The Rules of the Evolutionary Game for RNA:

A Statistical Characterization of the Sequence to Structure Mapping in RNA

DISSERTATION

zur Erlangung des
akademischen Grades

DOCTOR RERUM NATURALIUM

an der Formal- und Naturwissenschaftlichen Fakultät
der Universität Wien

vorgelegt von

Ivo Ludwig Hofacker

am

Institut für Theoretische Chemie

Mai 1994

An dieser Stelle möchte ich mich herzlich bei all denen bedanken, die zum entstehen der vorliegenden Arbeit beigetragen haben.

Da wäre zunächst Prof. Peter Schuster, der mir nicht nur Gelegenheit zur Dissertation gab, sondern auch für die ideellen wie materiellen Resourcen an diesem Institut sorgte. Er war stets ein interessanter Diskussionspartner und förderte gleichzeitig das eigenständige Arbeiten seiner Mitarbeiter.

Große Mitschuld an diesem Werk trägt auch Dr. Peter Stadler, der immer Zeit für Diskussionen und Ratschläge hatte. Ihm verdanke ich viele der hier gezeigten Ideen und viele nützliche TEX-sourcen. Er war immer ein Vorbild an Produktivität.

Dr. Walter Fontana verdanke ich wichtige Anregungen, er half mir immer wieder meine Ergebnisse in einen größeren Rahmen zu stellen.

Dr. Tomáš Kovář half mir von den ersten Schritten bis zu den tieferen Mysterien im Umgang mit UNIX.

Viele der für die Motivation so wichtigen Institutsfeste wurden von Dr. Christian Forst organisiert.

Vielen Dank auch allen Freunden im Institut und seinem Umfeld, die sowohl mit wissenschaftlichen Diskussionen als auch viel Ablenkung die letztem Jahre bereichert haben, insbesondere Mag. Thomas Griesmacher, Dr. Manfred Tacker, Mag. Erich Bauer, Mag. Herbert Kratky, Dr. Robert Hecht, Mag. Robert Happel, Dr. Andrea Gaunersdorfer und Dr. Wolfgang Schnabl.

Danken möchte ich auch der heimlichen Leitung des Instituts, Judith Jakubetz, die mir über alle bürokratischen Hürden hinweghalf.

Nicht zuletzt danke ich meinen Eltern, ohne deren Unterstützung mein Studium nicht möglich gewesen wäre.

Abstract

RNA folding is viewed here as a map assigning secondary structures to sequences and hence as an example of general genotype to phenotype mappings.

RNA secondary structures can be represented as planar vertex-labeled graphs or as trees. We construct recursion formulae enumerating various sub-classes of these graphs as well as certain structural elements and derive first order asymptotics for their frequencies. The number of secondary structures at fixed chain length turns out to be much lower than the number of sequences.

A package of efficient algorithms for the prediction of RNA secondary structures and their comparison is introduced. It is complemented by a novel heuristic "inverse folding" algorithm that searches for sequences which fold into a given secondary structure.

The mapping from sequences to secondary structures is then studied in a series of computer experiments. Statistics of secondary structure elements of folded random sequences are compiled and compared to random structures. In addition, we derive a lower bound on the number of folded structures from the mean number of base pairs. The frequency distribution of structures is found to be highly nonuniform, with few common structures and many rare ones, to a good approximation the distribution can be described by a generalized form of Zipf's law. Using the inverse folding algorithm sequences folding into some given structure are shown to be randomly distributed in sequence space.

As a consequence of the many to one mapping, common structures can be found within a relatively small distance (less than 20% of the diameter of sequence space) from any random sequence. Furthermore, the high number of neutral mutations leads to extended networks of neighboring sequences sharing the same structure. Neutral nets belonging to different structures can be found very close to one another. These properties can coexist only because of the high dimensionality of sequence space.

Although the RNA folding map induces *rugged* landscapes, the above properties make it ideally suited for evolutionary optimization. A structure with the desired properties is never too far away, once an acceptable solution has

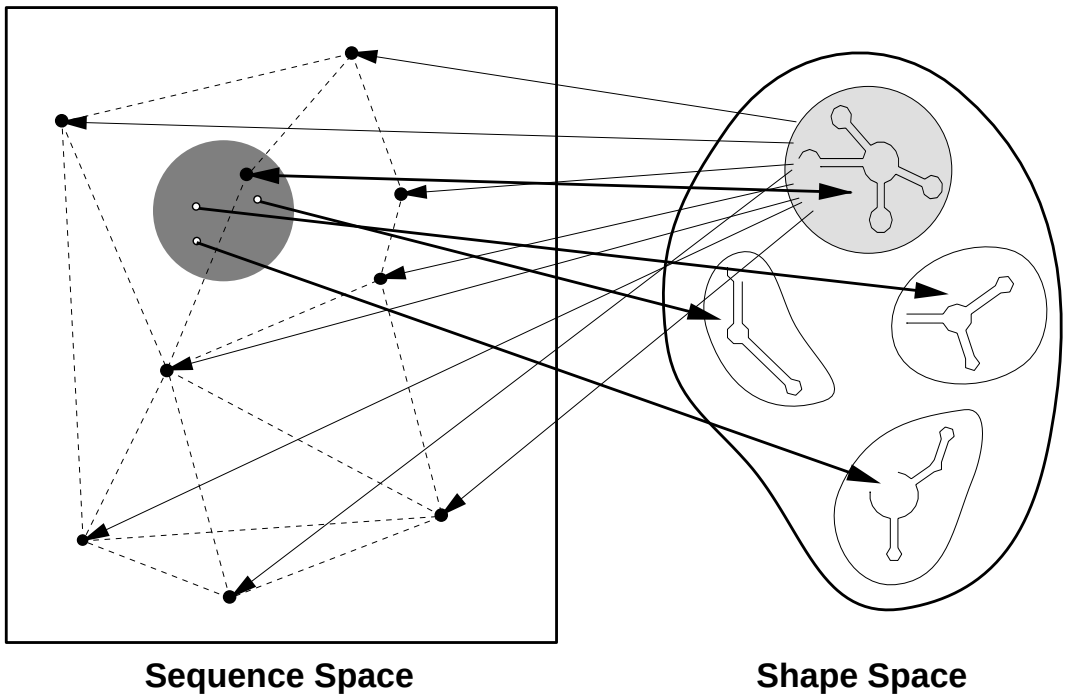


Figure 1: Schematic view of the RNA folding map. A relatively small ball in sequence space already contains sequences folding into most common structures. Sequences folding into a particular structure can be found anywhere in sequence space. Such sequences are connected by extended nets of structurally neutral neighbors.

been found the population can spread along the neutral net to distant parts of sequence space.

Kurzfassung

Die Faltung von RNA Molekülen wird in dieser Arbeit als eine Abbildung betrachtet werden, die jeder Sequenz eine Sekundärstruktur zuordnet. Sie stellt damit ein Beispiel für Genotyp Phänotyp Abbildungen im allgemeinen dar.

RNA Sekundärstrukturen können als als planare Graphen oder als Bäume dargestellt werden. Die Anzahl solcher Graphen und ihrer Strukturelemente bei fester Kettenlänge läßt sich rekursiv berechnen, aus den Rekursionen werden dann auch die asymptotischen Häufigkeiten hergeleitet. Dabei stellt sich heraus das die Zahl der Sequenzen sehr viel schneller wächst als die Zahl der Strukturen.

Im weiteren werden Algorithmen für die Vorhersage und den Vergleich von RNA Sekundärstrukturen diskutiert. Diese Algorithmen wurden zusammen mit einem neuartigen "inversen Faltungsalgorithmus" in einer Programmbibliothek implementiert. Bei der inversen Faltung werden in einer heuristischen Suche Sequenzen konstruiert die in eine vorgegebene Struktur falten.

Die weitere Untersuchung der Sequenz Struktur Abbildung geschieht in einer Reihe von Computerexperimenten. Eine Statistik der Sekundärstrukturelemente in gefalteten Strukturen wird erstellt und mit zufälligen Strukturen verglichen. Betrachtet man die Häufigkeiten von Strukturen, findet man eine sehr ungleichmäßige Verteilung die einige wenige sehr häufige dafür viele extrem seltene Strukturen aufweist. Die Verteilung kann gut durch eine allgemeine Form des aus der Textanalyse bekannten Zipf'schen Gesetzes beschrieben werden. Mit Hilfe der inversen Faltung wird gezeigt, daß die Urbilder einer Struktur im Seqenzraum gleichmäßig verteilt sind.

Als Folge dessen stellt sich heraus, daß typische Strukturen in relativ kleinem Abstand von einer beliebigen Anfangssequenz gefunden werden können (weniger als 20% des Durchmessers des Sequenzraums). Gleichzeitig, führt die hohe Anzahl an neutralen Mutationen zu ausgedehnten Netzwerken benachbarter Sequenzen mit gleicher Struktur. Zu verschiedenen Strukturen gehörige neutral Netze sind im Sequenzraum nur durch wenige Punktmutationen getrennt. Nur durch die hohe Dimension des Sequenzraums sind die letztgenannten Eigenschaften miteinander vereinbar.

Obwohl RNA Faltung zu rauhen Landschaften mit geringer Korrelation führt, ist sie durch die obigen Eigenschaften für evolutionäre Optimierung ideal geeignet. Eine Struktur mit den benötigten Eigenschaften läßt sich in einer vergleichsweise kleinen Umgebung finden, und sobald eine akzeptable Lösung gefunden ist kann sich die Population entlang der neutralen Netze ausdehnen um weit entfernte Teile des Sequenzraums zu erkunden.

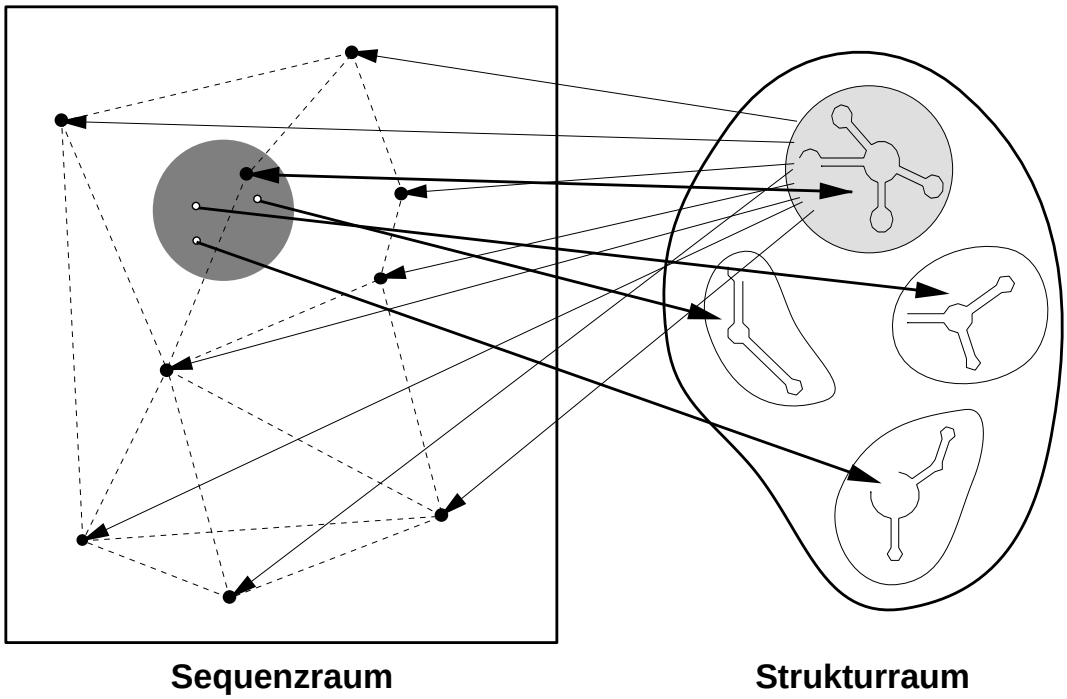


Figure 2: Schematische Darstellung der Abbildung von RNA Sequenzen auf Sekundärstrukturen. Eine relativ kleine Umgebung im Sequenzraum enthält bereits fast alle häufigen Strukturen. Sequenzen, die in eine vorgegebene Struktur fallen sind über den gesamten Sequenzraum verteilt zu finden. Diese Sequenzen sind durch ausgedehnte Netzwerke von neutralen Nachbarn miteinander verbunden.

1. Introduction

1.1. General Context

In his famous book "*The Origin of Species*" (1859) Charles Darwin put forward the first empirical theory of biological evolution in which he suggested that the diversity and complexity of present day organisms can be explained on the basis of just two key principles: inheritable *variation* and natural *selection*. While his theory quickly became one of the most influential contributions to natural science, it also remained most controversial for a long time.

One severe problem in Darwin's time was the ignorance of the laws and mechanisms of variation. Although Gregor Mendel had already formulated his laws of particulate inheritance their importance for evolution was not recognized. More than a century later the molecular basis of life has become clear. The sources of variation, in the form of mutation and recombination, are now understood and the knowledge of typical mutation rates allows us to estimate the rate of evolutionary change in the absence of selection. The concept of fitness, the basis of selection, however, has remained elusive. Traditionally, fitness has often been defined as the number of surviving offsprings. Such an a posteriori definition is clearly unsatisfactory and has led to the well known criticism that Darwin's "survival of the fittest" reduces to the tautology "survival of the survivor". At present, a prediction of fitness values still remains impossible, even for the simplest systems.

In 1932 Sewall Wright [2] introduced the concept of a fitness landscape ("adaptive landscape") assigning a numerical fitness value to every possible phenotype which he envisioned as a real valued vector of physical properties. Evolution could then be viewed as a walk through this space of phenotypes trying to optimize fitness. In fact, if such a mapping could be constructed over the space of genotypes, the evolutionary process would be reduced to a mathematically well defined optimization problem, such as Manfred Eigen's

quasi-species model [3, 4]. For this reason and because of the analogy to combinatorial optimization problems in computer science, this view of evolution has recently become very popular [5, 6, 7, 8, 9].

Since fitness is evaluated at the phenotype level, while mutation works on the genotype level, understanding of the genotype-phenotype mapping is crucial to the study of realistic fitness landscapes. Conversely, this mapping alone may yield sufficient information to predict generic features of the fitness landscapes that are built upon it. This thesis, therefore, does not attempt to propose a (necessarily artificial) model for fitness, but rather tries to study a (hopefully realistic) genotype-phenotype mapping.

The notion of fitness landscapes is not accepted without critique. In particular, it assumes a static environment and neglects the interaction between different species. Although the concept can be extended to allow for a varying environment [10] or even coevolving species [11], it remains best suited for models without interaction such as self replicating biopolymers.

1.2. Why RNA?

RNA provides an ideal, currently the only, tractable system to study genotype-phenotype relationships. Following Sol Spiegelman the phenotype for an RNA molecule can be defined as its spatial structure. RNA, thus, combines genotype and phenotype in the same molecule. Spiegelman's serial transfer experiments [12, 13, 14, 15, 16, 17, 18] have clearly shown that the structure of the RNA is the essential quantity selected for.

Furthermore, since the work of Thomas Cech [19, 20, 21] RNA is known to exhibit catalytic activity. While the activity of these so called "ribozymes" is usually restricted to cleavage and splicing of RNA itself, recent evidence suggests that RNA also plays a predominant role in ribosomal translation. These discoveries have given much support to the idea that an "RNA World" [22, 23] stood at the origin of life, in which RNA served both as carrier of genetic information as well as catalytically active substance. While RNA may not necessarily have been the first step in prebiotic evolution the idea

that RNA preceded not only DNA but also the invention of the translational system is widely accepted.

In spite of their, compared to proteins limited, repertoire of catalytic functions ribozymes are also gaining importance for biotechnological applications. The reason is that RNA is well suited for methods of "irrational design". Synthesis of random or partly randomized nucleotide sequences is nowadays standard procedure, as is their in vitro amplification by polymerase chain reaction (PCR). This allows selection experiments in which large ($10^{13} - 10^{15}$ molecules) pools of random RNA sequences are screened for some desired functions [24, 25, 26] or, in cases where the desired function can be encoded in a selection constraint, elegant in vitro evolution experiments [27, 28] with alternating cycles of selection and amplification by PCR under mutagenic conditions.

Yet another important, if pragmatic, reason to study RNA sequence structure relationship is the availability of algorithms for structure prediction. In spite of considerable efforts prediction of protein structures is, at the moment, both too unreliable and too costly for systematic studies. The same is true for attempts to model the three-dimensional structure of RNA although progress is being made in that direction [29, 30]. On the other hand efficient algorithms for the prediction of RNA secondary structure have been available for some time. While the accuracy of these algorithms may still be not quite satisfactory it is sufficient for the kind of investigation presented here, as our emphasis is not in the prediction of specific structures but the statistics of the folding. In contrast to proteins where secondary structure describes only local features, RNA secondary structure can be seen as a useful coarse grained picture of the the full spatial structure.

1.3. Combinatory Maps and Sequence Space

In this work the folding of RNA is viewed as a mapping from *sequence space* into the space of all possible structures a so-called *shape space*. The notion of a shape space was used previously in theoretical immunology for the set of all structures presented by all possible antigens [31, 32] The concept of

sequence space was first used in coding theory [33]. Such a map is called a combinatory map, a generalization of the landscape concept to non scalar entities. A combinatory map is defined [34] as a quintuple $(\mathcal{X}, d_{\mathcal{X}}; \mathcal{Y}, d_{\mathcal{Y}}; f)$ where $\mathcal{X}$ and $\mathcal{Y}$ are sets endowed with metrics $d_{\mathcal{X}}$ and $d_{\mathcal{Y}}$, respectively, and f is a mapping $\mathcal{X} \longrightarrow \mathcal{Y}$. For $\mathcal{Y} = \mathbb{R}$ and $d_{\mathcal{Y}}(a, b) = |a - b|$ we have a conventional landscape.

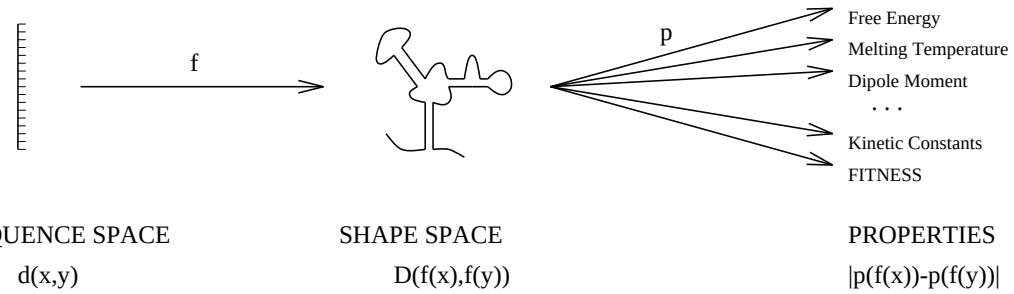


Figure 3: Scheme of RNA folding.

The canonic metric $d_{\mathcal{X}}$ for sequence space is the so called Hamming distance, given by the number of digits in which two aligned sequences differ. The Hamming metric is therefore tantamount to the minimum number of point mutations necessary to convert one sequence into another. A configuration space like this can also be represented as an undirected graph Γ, such that the distance between two points coincides with the minimum number of edges on Γ that have to be traversed to connect the two vertices. For binary sequences of length n the only possible choice for that graph is the n-dimensional hypercube. Thinking of sequence space as such a high-dimensional graph helps to keep in mind some basic properties:

While the number of possible sequences grows exponentially as κ^n, where κ is the number of letters in the alphabet, the maximal distance is only n.
The number of sequences in distance d is $p(d) = \frac{(\kappa-1)}{\kappa^n}\binom{n}{d}$ and therefore increases exponentially for $d \ll n$.
There are $d!$ shortest paths leading to a sequence in distance d.

In contrast to this the shape space is highly irregular. Even the number of nearest neighbors is variable. Note however, that the topology of the sequence space will not stay as simple when more complicated mutation

operators, such as insertions and deletions, are allowed. Although insertions and deletions play an important role in biological evolution they are much less frequent. For the sake of simplicity we will therefore only consider point mutations for the remainder of this work.

1.4. Organization of this Work

In the next chapter we will give a characterization of shape space for RNA secondary structures without reference to folding processes. Secondary structures will be defined and possible distance measures discussed. Furthermore, we derive recursions and asymptotics for the number of structures and certain structure properties.

Chapter 3 discusses the various algorithms used for RNA structure prediction. Efficient implementations of the folding algorithms based on dynamic programming are presented including a version for parallel computers with distributed memory. Furthermore, we introduce a new heuristic inverse folding algorithm that allows to search for sequences with a predefined structure. These algorithms provide the necessary tools for an exploration of sequence structure relationships through computer experiments.

Chapter 4 describes computational results for the RNA mapping. The frequency distribution of secondary structures is shown to be highly nonuniform following a form of Zipf's law with few very frequent and many rare structures. The more frequent structures are shown to be randomly distributed over sequence space. Statistics for structure elements of folded random sequences are presented and compared to those of random structures as defined in chapter 2. An analysis of correlation lengths shows that RNA folding leads to rough landscapes similar to many combinatorial optimization problems. A closer look, however, reveals optimization on the RNA landscape as being easier than expected: Typical structures can be found on the average in a distance of only about 20% of the diameter of sequence space. Moreover, the sequence space contains large *neutral networks*, i.e. connected sets of sequences that share the same structures. These nets can span the whole sequence space. An optimization process, therefore, will not get stuck in local optima too easily.

The results are discussed in chapter 5.

2. RNA Secondary Structures

Much like DNA, RNA can form stable double helices of complementary strands. Since RNA usually occurs single stranded, formation of double helical regions is accomplished by the molecule folding back onto itself to form Watson-Crick (**G**≡**C** and **A**=**U**) base pairs or the slightly less stable **G**−**U** pairs. This process is the major driving force for RNA structure formation. Other, usually weaker, intermolecular forces and the interaction with the aqueous solvent shape its spatial structure.

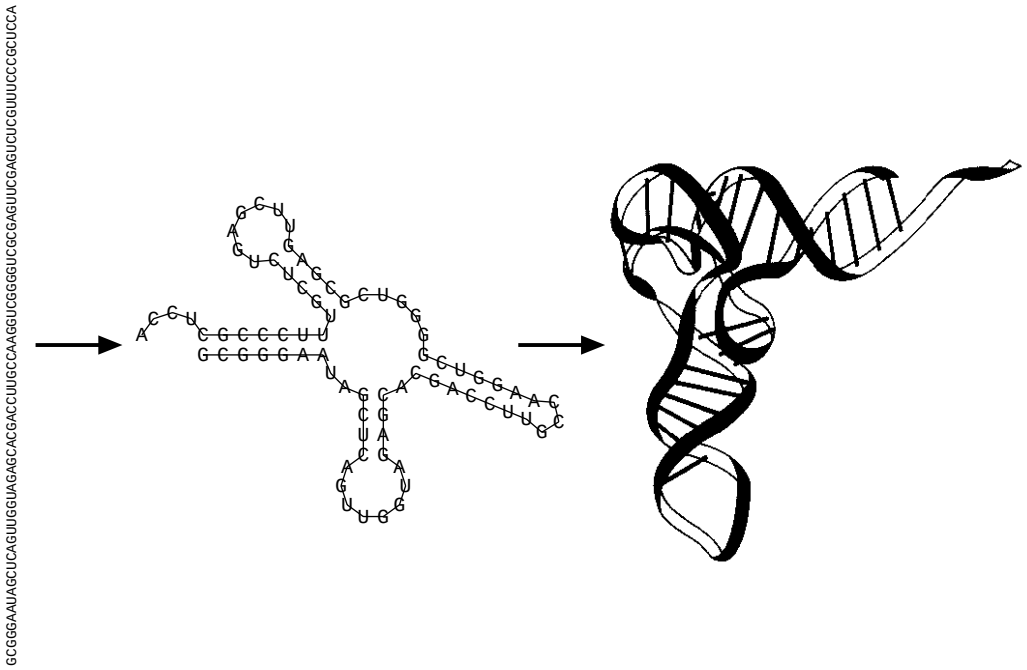


Figure 4: Folding of an RNA sequence into its spatial structure. The process is partitioned into two phases: in the first phase only the Watson-Crick-type base pairs are formed (which constitute the major fraction of the free energy), and in the second phase the actual spatial structure is built by folding the planar graph into a three-dimensional object. The example shown here is phenylalanyl-transfer-RNA (t-RNA$^{\mathrm{phe}}$) whose spatial structure is known from X-ray crystallography.

RNA folding can therefore be conveniently partitioned into two steps, the first being the formation of base pairs yielding the so-called secondary structure that can be represented as a planar graph, and secondly the folding of this

graph into a 3D structure. In this work we will only be concerned with the first step.

It may at first seem questionable whether secondary structure provides an adequate level of description for an RNA molecule. One should, however, keep in mind that when talking of an object's structure we do not imagine a list of coordinates for each of its constituents, but rather a number of relationships between them, yielding enough information to understand the objects crucial properties. Such a representation necessarily implies some degree of coarse graining and different problems may require different levels of resolution.

RNA secondary structures provide such a coarse graining of a folded RNA molecule that is useful for several reasons.

- Conventional base pairing and base pair stacking cover the major part of the free energy of folding.
- Secondary structures are used successfully in the interpretation of RNA function and reactivity.
- Secondary structures are conserved in evolutionary phylogeny.

At the same time this representation is very convenient:

- Secondary structures are discrete and therefore easy to compare.
- They are easy to visualize since they're planar graphs.
- Efficient methods exist for the computation of secondary structures.

In the following we will give a formal definition of secondary structures as graphs. Note that our definition, somewhat arbitrarily, ranks pseudo-knots as a tertiary interaction. Although pseudo-knots seem to be important for biological function [35] their inclusion would complicate the mathematical and computational treatment unduly.

2.1. Secondary Structure Graphs

2.1.1. Definitions

Definition 1.1. [36] A *secondary structure* is a vertex-labeled graph on n vertices with an adjacency matrix A fulfiling

(1) $a_{i,i+1} = 1$ for $1 \leq i < n$;

(2) For each i there is at most a single $k \neq i-1, i+1$ such that $a_{ik} = 1$;

(3) If $a_{ij} = a_{kl} = 1$ and $i < k < j$ then $i < l < j$.

We will call an edge (i, k), $|i - k| \neq 1$ a bond or base pair. A vertex i connected only to $i - 1$ and $i + 1$ will be called unpaired. Condition (3) assures that the structure contains no pseudo-knots.

A vertex i is said to be *interior* to the base pair (k, l) if $k < i < l$. If, in addition, there is no base pair (p, q) such that $p < i < q$ we will say that i is *immediately interior* to the base pair (k, l). A base pair (p, q) is said to be (immediately) interior if p and q are (immediately) interior to (k, l).

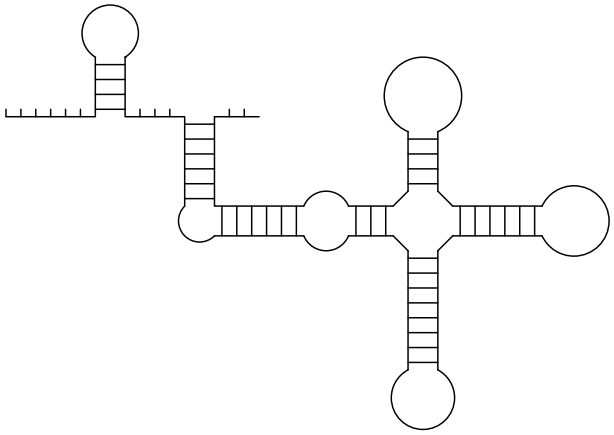


Figure 5: Example of a secondary structure graph.

Definition 1.2. A secondary structure consists of the following structure elements

(1) A *stack* consists of subsequent base pairs $(p-k, q+k)$, $(p-k+1, q+k-1)$, ..., (p, q) such that neither $(p-k-1, q+k+1)$ nor $(p+1, q-1)$ is a base pair. $k+1$ is the *length* of the stack, $(p-k, q+k)$ is the terminal

base pair of the stack. Isolated single base pairs are considered as stacks as well.

(2) A *loop* consists of all unpaired vertices which are immediately interior to some base pair (p, q), the "closing" pair of the loop.

(3) An *external vertex* is an unpaired vertex which does not belong to a loop. A collection of adjacent external vertices is called an external element. If it contains the vertex 1 or n it is a free end, otherwise it is called joint.

If a stack ends in a base pair (p, q) with no unpaired vertices immediately interior to it we speak of a loop with size zero.

Lemma 1.3. Any secondary structure $\mathcal{S}$ can be uniquely decomposed into stacks, loops, and external elements.

Proof. Each vertex which is contained in a base pair belongs to a unique stack. Since an unpaired vertex is either external or immediately interior to a unique base pair the decomposition is unique: Each loop is characterized uniquely by it "closing" base pair.

Definition 1.4. A stack $[(p, q), \ldots, (p+k, q-k)]$ is called *terminal* if $p-1 = 0$ or $q + 1 = n + 1$ or if the two vertices $p - 1$ and $q + 1$ are not interior to any base pair. The sub-structure enclosed by the terminal base pair (p, q) of a terminal stack will be called a *component* of $\mathcal{S}$. We will say that a structure on n vertices has a terminal base pair if $(1, n)$ is a base pair.

Lemma 1.5. A secondary structure may be uniquely decomposed into components and external vertices. Each loop is contained in a component.

The proof is trivial. Note that by definition the open structure has 0 components.

Definition 1.6. The degree of a loop is given by 1 plus the number of terminal base pairs of stacks which are interior to the closing bond of the loop. A loop of degree 1 is called *hairpin (loop)*, a loop of a degree larger than 2 is called *multi-loop*. A loop of degree 2 is called *bulge* if the closing pair of the loop and the unique base pair immediately interior to it are adjacent; otherwise a loop of degree 2 is termed *interior loop*.

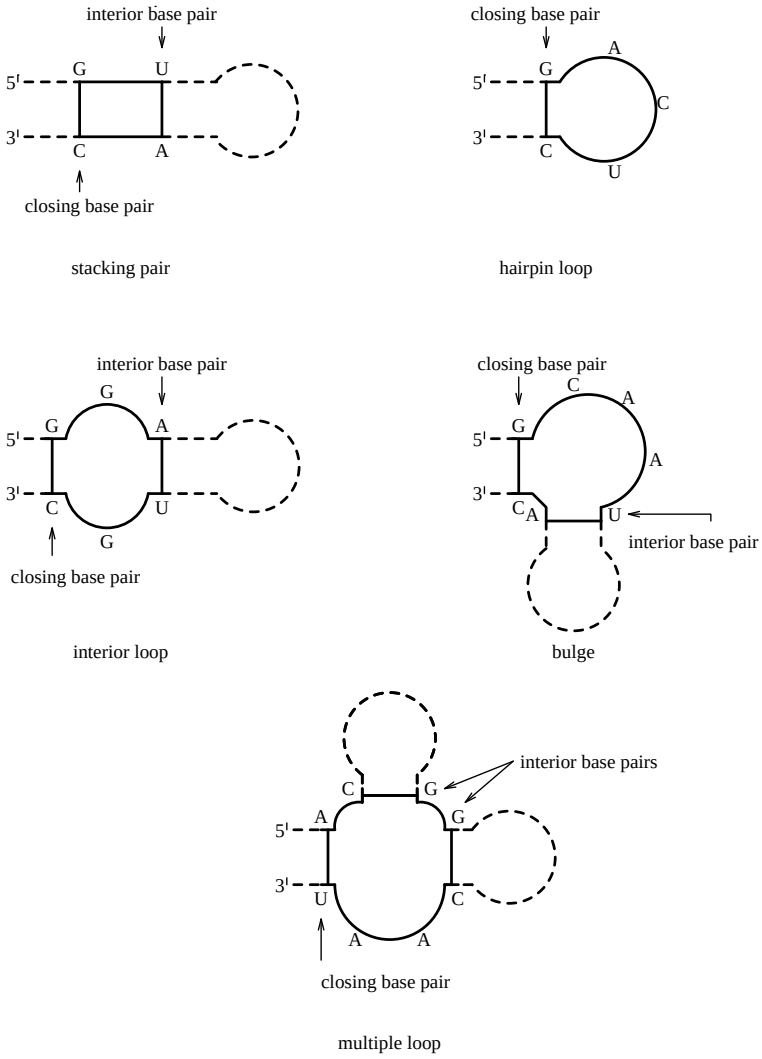


Figure 6: Basic structure elements. Every secondary structure can be decomposed into such basic elements.

It is often useful to lump loops of all degrees together into one class and to consider, for example, the total number of loops

$$n_{\mathrm{L}} \;=\; n_{\mathrm{H}} \,+\, n_{\mathrm{B}} \,+\, n_{\mathrm{I}} \,+\, n_{\mathrm{M}}$$

which must be identical to the number of stacks, $n_{\mathrm{L}} = n_{\mathrm{S}}$.

2.1.2. Representation of Secondary Structures

A particularly easy way to draw secondary structure graphs as defined above was suggested by Ruth Nussinov. The bases of the sequence are placed equidistant to on another on a circle and for each base pair a chord is drawn between the two bonded bases. Since the structures are un-knotted by definition, no two chords will intersect.

A string representation $\mathbf{S}$ can by obtained by the following rules:
(1) If vertex i is unpaired then $\mathbf{S}_i =$ "."
(2) If (p, q) is a base pair and $p < q$ then $\mathbf{S}_p =$ "(" and $\mathbf{S}_q =$ ")"

These rules yield a sequence of matching brackets and dots (*cf.* 8C) called *bracket notation.*

Paulien Hogeweg and Danielle Konings conceived a related graphical method for the comparison of RNA secondary structures called *mountain representation* [37, 38, 39] by identifying "(", ")", and ".", with "up", "down", and "horizontal", respectively.

The bracket notation above implies an equivalent representation as trees. A secondary structure $\mathcal{S}$ can be translated into a rooted ordered tree (linear tree) Υ by representing a base pair (p, q) by a node x such that the sons $y_1, \ldots y_k$ of x correspond to the base pairs $(p_1, q_1) \ldots (p_k, q_k)$ immediately interior to (p, q) [40]. For each unpaired vertex z a half-node (leaf) is added to the node representing the closing pair of the loop containing z. An additional node is added as the *root* of the tree that is the father of all nodes representing external digits and terminal base pairs. This assures that secondary structures with free end are not represented by a forest. For details see table 1, an example is given in figure 7. Other coarse grained tree representations where already suggested by Zuker and Sankoff [41] and Bruce Shapiro [42].

The latter representation makes explicit that the folding process can be viewed as a map between linear and nonlinear combinatorial structures: sequences and trees.

Waterman's definition of secondary structures implies that each branch of the corresponding tree representation Υ has least one terminal half-vertex, or equivalently, each matching pair of brackets contains at least one $\circ$. In

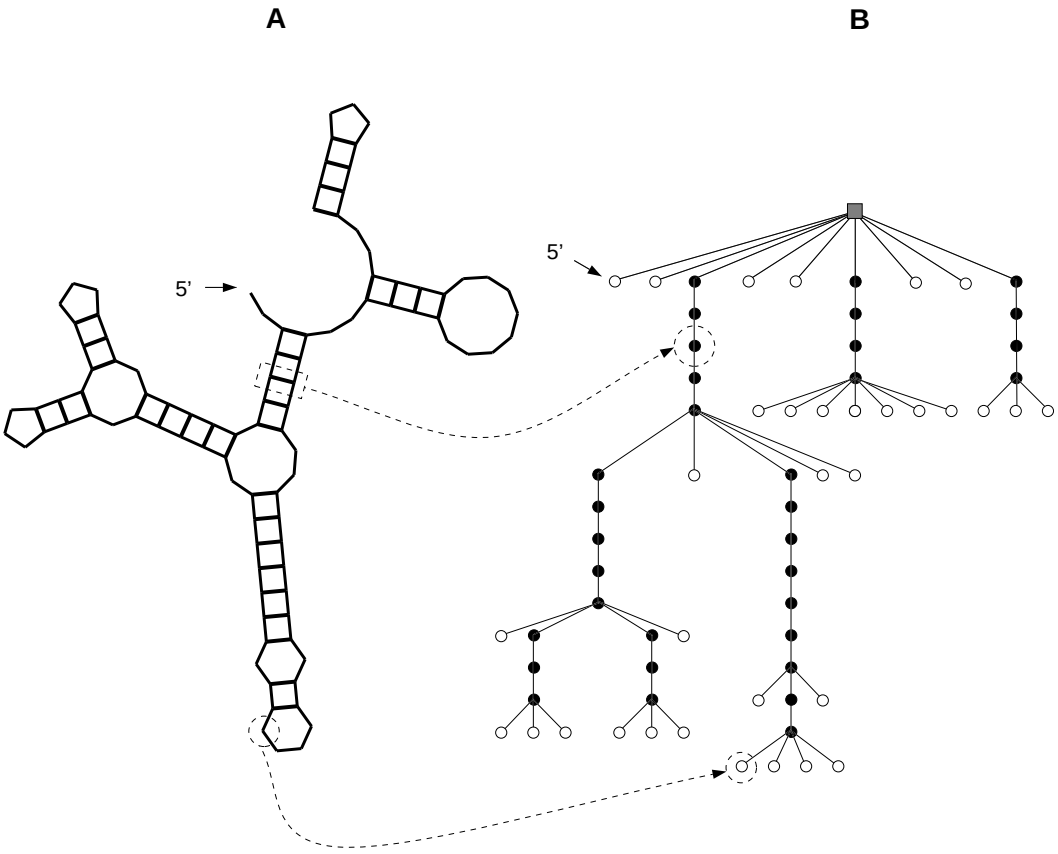


Figure 7: A secondary structure graph (**A**) is equivalent to an ordered rooted tree (**B**). An internal node (black) of the tree corresponds to a base pair, a leaf node (white) corresponds to one unpaired nucleotide, and the root node (black square) is a virtual parent to the external elements. Contiguous base pair stacks translate into "ropes" of internal nodes and loops appear as bushes of leaves.

biological applications the number of unpaired positions is at least 3, implying at least 3 unpaired positions within each pair of matching brackets. From the combinatorial point of view it makes perfect sense to consider the general problem with a minimum number $m \geq 0$ of unpaired vertices in each hairpin loop. In fact, for $m = 0$ one recovers three well known Motzkin families [43, 44].

A secondary structure tree Υ can be rewritten as *homeomorphically irreducible trees* (HITs) which will be denoted by $\mathcal{H}$, by merging internal nodes corresponding to the same stack or to consecutive unpaired bases into on

Table 1. Interconversion of secondary structures and trees.

A secondary structure graph (SSG) is converted into a tree graph (TG) by the following procedure:
(1) Assign to each unpaired base a leaf node, and to each base pair (two SSG-nodes) one internal TG-node.
(2) Each internal TG-node corresponding to a base pair (k,l) is father to all nodes corresponding to bases or pairs immediately interior to (k,l). The virtual node representing the root of the tree is father to all nodes corresponding to external bases and terminal base pairs.
(3) Order siblings according to their corresponding position in the sequence and connect each node to its parent.

A TG is converted into a SSG by the following procedure:
(1) Replace each internal TG-node by a connected pair of SSG-nodes (base pair). The TG-edges are inherited by the left SSG-node of the pair.
(2) For any TG-node with a left sibling replace the edge to its parent with an edge to its left sibling.
(3) An SSG-node that is base paired has three edges (two from the backbone, and one from the pairing), otherwise it has two. 5'- and 3'-ends have one less. Complete the SSG by inserting all missing edges into a node as connections to the corresponding parent in the TG, proceeding from deep to shallow levels.

node. The apparently simpler tree structure of the HIT is compensated by the assignment of weights (w) to the internal nodes and leaves. A weight reflects the number of nodes or leaves in the full tree Υ which are lumped into a single node or leaf in the HIT representation. The transformation from the full tree to the HIT retains complete information on the structure. Secondary structure graph, full tree, HIT, and the linear representations (like the mountain representation M) are equivalent.

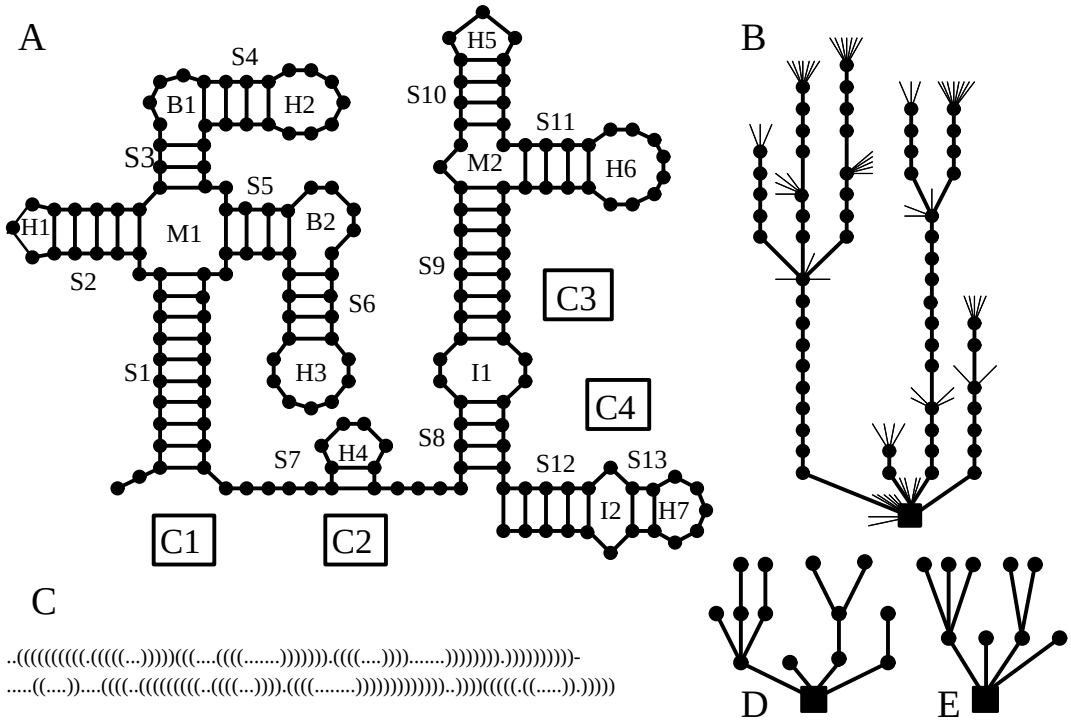


Figure 8: Representations of secondary structures. The notation **A** is common in biology. Structure elements are indicated as follows: H hairpin loops, I interior loops, B bulges, M multi-loops, S stacks. The structure consists of four components, indicated as *C1* through *C4*.
B is the corresponding full tree notation Υ, and **C** is the corresponding linear bracket representation.
D is a coarse grained representation Σ or T obtained from **B** by contracting each stack to a single vertex and omitting the half-vertices (leafs) representing the unpaired positions. **E** is the homeomorphically irreducible tree H obtained from **D**.

2.1.3. Coarse Graining of Secondary Structures

For some applications it is useful to work with even more simplified representations. Various coarse grained representations have been proposed. A coarse grained tree $\mathcal{T}$, the *loop structure*, is obtained by denoting a stack by a single vertex and omitting the unpaired bases. Representing an entire stack instead of a base pair by a single vertex means in terms of the full tree representation Υ that each vertex of degree 2 not carrying a leaf (half-vertex), except for the root, is merged with its son and *then* the leafs are removed (*cf.* fig. 8D). The number of vertices in $\mathcal{T}$ equals the number of stacks in $\mathcal{S}$, the

number of components of $\mathcal{S}$ coincides with the number of sons of the root in $\mathcal{T}$.

The representation Σ proposed by Shapiro [42] is obtained by additionally labeling each vertex with the type of the loop in which the corresponding stem ends: hairpin loop (H), bulge (B), interior loop (I), multi-loop (M). The virtual root is labeled (R). Furthermore, stacks are represented by vertices of degree 2, labeled (S), which are inserted as fathers of the corresponding loops. The nodes may be assigned weights giving the size of the loop or stack, respectively. If the weights are omitted Σ is equivalent to $\mathcal{T}$.

A slightly more accurate representation of a secondary structure, $\mathcal{H}'$, is obtained from the HITs by simply omitting the weights. A gradual coarse graining can be obtained by removing vertices with weights smaller than some threshold.

A strongly coarse grained representation, the *branching structure*, H is the homeomorphically irreducible tree obtained form $\mathcal{T}$by removing all vertices of degree 2 (except for the root) and all leafs in $\mathcal{T}$. Again the number of components of $\mathcal{S}$ equals the number of sons of the root.

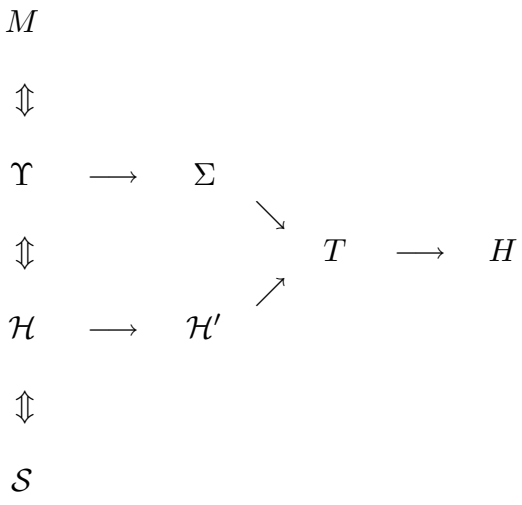


Figure 9: Relation of various representations of secondary structures. $\Longleftrightarrow$ denote equivalent representations, $\longrightarrow$ means a coarse graining.

Definition 1.7. Let $\mathcal{S}$ be an arbitrary secondary structure. For all $\mathcal{S}$ let us denote by $\Omega(\mathcal{S})$ the unique secondary structure which is obtained from $\mathcal{S}$ by the following procedure:

1) For each hairpin, open its stack and add the corresponding bases to the hairpin loop.
2) If a bulge or interior loop follows, then add its digits also to the hairpin and continue by opening its stack.
3) If a multi-loop or a joint follows, then add the now unpaired digits to the multi-loop and stop.

Waterman (1978) used the above procedure to define the order $\omega(\mathcal{S})$ of a secondary structure as the smallest number of repetitions of Ω necessary to obtain the open structure. Of course, the open structure has order $\omega = 0$ and any structure without a multi-loop has order $\omega = 1$. Waterman's degree ω, defined below, coincides with the height of H (*cf.* fig. 8E).

2.2. Comparison of Secondary Structures

An important prerequisite for our study is a suitable distance measure on the space of possible secondary structures. Several possibilities exist to define such a metric. One of the simplest possibilities is the base pair distance. It is given by the number of base pairs (i, j) present in only one of the two structures being compared or, in other words, the minimum number of base pairs that have to be opened or closed to convert one structure into the other. This measure is best suited to compare different structures on the same sequence. Since opening and closing of single base pairs can be thought of as the elementary steps in re-folding an RNA molecule, the base pair distance measures the likeliness of such a re-folding. On the other hand structures that seem very similar to the eye, can have a large base pair distance for instance if the size of a hairpin differs by one base.

Another frequently used method uses conventional string alignment on the dot and bracket encoding of secondary structures [37, 38]. In most cases such alignment distances yield good results, however, the two matching brackets encoding a base pair are treated as independent, yielding sometimes unintuitive results.

Our preferred method are therefore tree edit distances. This technique is well known in computer science [45], it's application to RNA secondary structure was first proposed by Bruce Shapiro [46]. As usual the task is to find a sequence of editing steps such as to transform a tree $\mathcal{T}_1$ into a tree $\mathcal{T}_2$ with minimal cost. The allowed edit operations here are deletion, insertion and (sometimes) relabelling of a node and the cost of an editing sequence is given by the sum of the costs of the individual editing operations taken from a cost table. An important advantage of this distance measure is that it can be used for all kinds of tree representations and, therefore, for all levels of coarse graining of secondary structures. One need only define an appropriate cost table for the edit operations. Computation of tree edit distances can again be done by a dynamic programming algorithm. It is, however somewhat more complicated than simple string editing which is, in fact, a special case of tree editing for tree consisting solely of leaves. The time complexity of the algorithm generally is $\mathcal{O}(|\mathcal{T}_1| \cdot |\mathcal{T}_2| \cdot L_1 \cdot L_2)$ where $|\mathcal{T}|$ is the number of nodes and L is the depth of the tree $\mathcal{T}$. Since the depth of a typical tree increases as the log of its size, tree editing is slower than sequence alignment by $\mathcal{O}(\log^2(n))$. Note that the typical size of secondary structure tree is proportional to but considerably smaller than the sequence length.

Note that all distances defined above induce a metric on the shape space of secondary structures, as do all "edit distances" as long as the cost matrix is positive and symmetric.

2.3. Enumeration of Secondary Structure Graphs

2.3.1. The Basic Recursion

A secondary structure on $n+1$ digits may be obtained from a structure on n digits either by adding a free end at the right hand end or by inserting a base pair $(1, k+2)$. In the second case the substructure enclosed by this pair is an arbitrary structure on k digits, and the remaining part of length

$n-k-1$ is also an arbitrary valid secondary structure. Therefore, we obtain the following recursion formula for the number S_n of secondary structures:

$$\begin{aligned} S_{n+1} &= S_n + \sum_{k=m}^{n-1} S_k S_{n-k-1}, \qquad n \geq m+1 \\ S_0 &= S_1 = \ldots = S_{m+1} = 1 \end{aligned} \tag{1}$$

Equ.(1) has first been derived by Waterman [36]; m denotes the minimum number of unpaired digits in a hairpin loop. Note that our definition of S_n differs from Waterman's for $n < m$: he used $S_n = 0$.

The above recursion can be used to develop an algorithm for generating random secondary structures with a uniform distribution

$$\text{Prob}\{\mathcal{S}\} = 1/S_n \tag{2}$$

in the *shape space* of all secondary structures over a given chain length. To construct random structures recursively just note that $S_k S_{n-k-1}/S_{n+1}$ gives the probability that a random structure of length $n+1$ consists of a random structure $\mathcal{S}'$ of length k enclosed by a base pair appended to a second random structure $\mathcal{S}$ of length $n-k-1$.

2.4. Recursions

2.4.1. Structures with Certain Properties

Let $J_n(b)$ denote the number of structures on n vertices with exactly b components. The derivation of the recursion relations parallels the argument leading to equ.(1):

$$\begin{aligned} J_{n+1}(b) &= J_n(b) + \sum_{k=m}^{n-1} S_k J_{n-k-1}(b-1), \qquad b > 0,\, n \geq m+1 \\ J_n(b) &= 0,\, b > 0, n \leq m+1, \qquad J_n(0) = 1, n \geq 0 \end{aligned} \tag{3}$$

because adding an unpaired digits to a structure on n digits does not change the number of components, while introducing an additional bracket makes

the bracketed part of length k a single component and does not affect the remainder of the sequence.

Let $H_n(b)$ denote the number of structures with exactly b base pairs (bonds) on n vertices. The recursion

$$\begin{gathered} H_{n+1}(b) = H_n(b) + \sum_{k=m}^{n-1} \sum_{\ell=0}^{b-1} H_k(\ell) H_{n-k-1}(b-\ell-1), \qquad b > 0,\, n \geq m+1 \\ H_n(b) = 0,\, b > 0, n \leq m+1, \qquad H_n(0) = 1, n \geq 0 \end{gathered} \tag{4}$$

is also immediate. One just has to observe that an additional sum over the number of unpaired digits in the newly bracketed part of the structure has to be introduced. This recursion has also been considered in [47]. Recently Schmitt and Waterman [48] obtained the closed expression $H_n(b) = \frac{1}{b}\binom{n-b}{b+1}\binom{n-b-1}{b-1}$ for the special case $m = 1$.

Analogously we obtain for the number $E_n(b)$ of structures with b external digits

$$\begin{gathered} E_{n+1}(b) = E_n(b-1) + \sum_{k=m}^{n-1} S_k E_{n-k-1}(b) \qquad b \geq 0,\, n \geq m+1 \\ E_n(n) = 1, \quad E_n(b) = 0 \quad b \neq n,\ n \leq m+1, \quad E_n(-1) = 0 \end{gathered} \tag{5}$$

It is a bit more tricky to obtain a recursion for number $N_n(b)$ of sequences with a given number of stacks. To this end we introduce an auxiliary variable $Z_n(b)$ denoting the number of secondary structures with exactly b stacks *given* that its $3'$ and $5'$ ends are paired. We obtain then

$$\begin{gathered} N_{n+1}(b) = N_n(b) + \sum_{k=m}^{n-1} \sum_{\ell=0}^{b} Z_{k+2}(\ell) N_{n-k-1}(b-\ell), \quad b > 0,\, n \geq m+1 \\ N_n(0) = 1, \quad N_n(b) = 0, \quad b > 0,\, n \leq m+1 \end{gathered} \tag{6}$$

For the auxiliary variably we find

$$Z_n(b) = Z_{n-2}(b) + N_{n-2}(b-1) - Z_{n-2}(b-1), \qquad Z_0(b) = Z_1(b) = 0 \tag{7}$$

by enclosing structures on $n-2$ digits by a base pair.

Let $A_n(b)$ denote the number of structures with exactly b hairpins. Since the number of hairpins is unchanged by enclosing a substructure which already contains a base pair in an additional base pair we get

$$\begin{aligned} A_{n+1}(b) &= A_n + \sum_{k=m}^{n-1} \left[\sum_{\ell=1}^{b} A_k(\ell) A_{n-k-1}(b-\ell) + A_{n-k-1}(b-1) \right] \\ & \qquad n \geq m+1 \\ A_n(b) &= \delta_{0,b} \qquad n \leq m+1 \end{aligned} \tag{8}$$

2.4.2. Structure Elements

The total number U_n of unpaired bases in the set of all structures can be obtained as follows: By adding an unpaired base to each structure on n digits we have the U_n unpaired digits present in them plus the S_n newly added ones. By introducing the base pair $(1, k+2)$ we have S_k times all the unpaired digits in the remainder of the sequence plus all the unpaired digits in the newly bracket part of length k times the the number of structures which can be formed from the remainder of the structure. Summing over k we find

$$\begin{aligned} U_{n+1} &= (U_n + S_n) + \sum_{k=m}^{n-1} [S_k U_{n-k-1} + S_{n-k-1} U_k], \qquad n \geq m+1 \\ U_n &= n, \quad n \leq m+1 \end{aligned} \tag{9}$$

Denote the total number of base pairs by P_n. It is clear that $2P_n + U_n = nS_n$. For sake of completeness we state the recursion for P_n:

$$\begin{aligned} P_{n+1} &= P_n + \sum_{k=m}^{n-1} \{S_k P_{n-k-1} + S_{n-k-1}(P_k + S_k)\} \\ P_n &= 0, \quad n \leq m+1 \end{aligned} \tag{10}$$

By an analogous reasoning we find for the total number I_n of components in the set of all secondary structures on n vertices.

$$\begin{aligned} I_{n+1} &= I_n + \sum_{k=m}^{n-1} S_k [I_{n-k-1} + S_{n-k-1}] \\ I_n &= 0 \quad n \leq m+1 \end{aligned} \tag{11}$$

The number N_{n+1} of stacks in the set of structures on $n+1$ digits consists of all stacks on n digits plus all stacks in the tail times the number of structures with the newly introduced base pair plus all stacks within the newly formed base pair times the number of structures in the tail. The newly formed base pair introduces an additional stack for all $S_k - S_{k-2}$ structures in its interior which do not have a terminal base pair. (For the S_{k-2} structures with terminal base pair a stack is elongated.) Therefore

$$\begin{aligned} N_{n+1} = N_n + \sum_{k=m}^{n-1} \left\{ S_k N_{n-k-1} + S_{n-k-1}(N_k + S_k) \right\} \\ - \sum_{k=m+2}^{n-1} S_{k-2} S_{n-k-1} \qquad\qquad n \geq m+1 \\ N_n = 0,\ n \leq m+1 \end{aligned} \tag{12}$$

Let $Q_n(b)$ denote the number of loops with b unpaired digits in the set of all secondary structures. For $n+1$ vertices we retain all loops from the set of loops on n digits by adding a vertex to the $3'$ end; additional we find all loops in the tail-substructure for each possible structure interior to the new base pair. The third contributions consists of all loops interior to the new base pair times all possible structures in the tail. A loop with b unpaired vertices remains unchanged and additionally each structure with exactly b external vertices within the new base pair gives rise to an additional loop with b unpaired digit.

$$\begin{aligned} Q_{n+1}(b) = Q_n(b) + \sum_{k=m}^{n-1} \left\{ Q_{n-k-1}(b) S_k + S_{n-k-1}[Q_k(b) + E_k(b)] \right\} \\ n \geq m+1,\ b > 0 \\ Q_n(b) = 0,\ n \leq m+1 \end{aligned} \tag{13}$$

For loops without unpaired digits the recursion is slightly different since structures without external digits within the new base pair do not provide a loop if they consist of a single component, i.e. if they end in base pair. There

are S_{k-2} such structures on k vertices.

$$\begin{aligned} Q_{n+1}(0) = Q_n(0) + \sum_{k=m}^{n-1} \left\{ Q_{n-k-1}(0) S_k + S_{n-k-1}[Q_k(0) + E_k(0)] \right\} \\ - \sum_{k=m+2}^{n-1} S_{n-k-1} S_{k-2} \quad n \geq m+1 \\ Q_n(0) = 0,\, n \leq m+1 \end{aligned} \tag{14}$$

Note that only multi-loops can have size zero.

Let $W_n(b)$ denote the number of stacks with exactly b base pairs in the set of secondary structures. For each structure in the bracketed part there are $W_{n-k-1}(b)$ stacks of suitable size while for each structure in the tail there only $W_k(b) - W_k^-(b) + W_k^+(b)$ such stacks; $W_k^-(b)$ denotes all stacks of correct length which are elongated by the new bracket and $W_k^+(b)$ denotes all stacks which are too short by one base pair and are elongated by the new pair. Clearly, $W_k^-(b)$ is just the number of structures on k digits with a terminal stack of length b, while $W_k^+(b)$ is the number of structures with a terminal stack of length $b-1$. We have

$$W_k^-(b) = \begin{cases} S_{k-2b} - S_{k-2b-2} & k > m+2b+1 \\ 1 & k = m+2b, m+2b+1 \\ 0 & k < m+2b \end{cases} \tag{15}$$

and $W_k^+(b) = W_k^-(b-1)$ for these auxiliary variables.

$$\begin{aligned} W_{n+1}(b) = \\ &= W_n(b) + \sum_{k=m}^{n-1} \left\{ W_{n-k-1}(b) S_k + S_{n-k-1}[W_k(b) - W_k^-(b) + W_k^+(b)] \right\} \\ &= W_n(b) + \sum_{k=m}^{n-1} \left[W_{n-k-1}(b) S_k - S_{n-k-1} W_k(b) \right] + \\ &\quad + \sum_{k=m+2b+2}^{n-1} S_{k-2b-2} S_{n-k-1} - 2 \sum_{k=m+2b}^{n-1} S_{k-2b} S_{n-k-1} + \\ &\quad + \sum_{k=m+2b-2}^{n-1} S_{k-2b+2} S_{n-k-1} \\ &\quad W_n(b) = 0 \quad \text{for } n \leq m+1 \end{aligned} \tag{16}$$

Let $L_n(d)$ denote the number of loops of degree d in the set of all secondary structures. By Y_n and B_n, respectively., we will denote the number of interior loops and bulges. Let us start with bulges and interior loops: Let X_n^* denote the number of structures that yield a bulge if included into an extra pair of brackets, and let X_n^{**} denote the number of structures that yield an interior loop if included into an extra bracket, i.e. the number of structures having a free end on both sides. Clearly $X_n^{**} = J_{n-2}(1)$, as structures with zero components would yield a hairpin while structures with more components would yield a multi-loop. In order to calculate X_n^* we observe that a bulge is formed by a new bracket if the structure enclosed has only a single component and ends neither in a base pair nor in free ends on both sides. As there are S_{n-2} structures resulting in a stack elongation if $n \geq m+2$ (and none otherwise) we have

$$X_n^* = J_n(1) - J_{n-2}(1) - S_{n-2} \quad n \geq m+2 \tag{17}$$

The recursions for loops of degree 2 are now straight forward:

$$\begin{aligned}
B_{n+1} &= B_n + \sum_{k=m}^{n-1} \left\{ S_k B_{n-k-1} + S_{n-k-1}[B_k + X_k^*] \right\} \\
Y_{n+1} &= Y_n + \sum_{k=m}^{n-1} \left\{ S_k Y_{n-k-1} + S_{n-k-1}[Y_k + J_{k-2}(1)] \right\} \\
L_{n+1}(2) &= L_n(2) + \sum_{k=m}^{n-1} \left\{ S_k L_{n-k-1}(2) + S_{n-k-1}[L_k(2) + J_k(1)] \right\} \\
&\quad - \sum_{k=m+2}^{n-1} S_{n-k-1} S_{k-2} \\
B_n = Y_n &= L_n(2) = 0 \qquad n \leq m+1
\end{aligned} \tag{18}$$

Hairpins are generated either by stack-elongation of a structure with a single hairpin or by enclosing the open structure into the additional bracket. Thus

$$\begin{aligned}
L_{n+1}(1) &= L_n(1) + \sum_{k=m}^{n-1} \left\{ S_k L_{n-k-1}(1) + S_{n-k-1}[L_k(1) + 1] \right\} \quad n \geq m+1 \\
L_n(1) &= 0 \quad n \leq m+1
\end{aligned} \tag{19}$$

For multi-loops finally we obtain the recursion

$$\begin{aligned} L_{n+1}(d) = L_n(d) + \sum_{k=m}^{n-1} \left\{ S_k L_{n-k-1}(d) + S_{n-k-1}[L_k(d) + J_k(d-1)] \right\} \\ \text{for} \quad d \geq 2,\, n \geq m+1 \\ L_n(d) = 0 \qquad \text{for } n \leq m+1 \end{aligned} \tag{20}$$

Summing over all loop degrees d we recover the recursion for the total number of stacks, since for each stack there is exactly one loop.

The total number of external digits, E_n, can be obtained directly as sum of the numbers $E_n(b)$. For sake of completeness we mention that it fulfills the recursion

$$\begin{aligned} E_{n+1} = E_n + S_n + \sum_{k=m}^{n-1} S_k E_{n-k-1} \quad n \geq m+1 \\ E_n = n \quad n \leq m+1 \end{aligned} \tag{21}$$

2.4.3. Secondary Structures of a Given Order

Let $D_n(c, \omega)$ be the number of secondary structures with c components and order ω. Furthermore let $D_n^*(\omega)$ be the number of structures which yield a structure of order ω when enclosed by an additional base pair. It is clear that the following recursion holds

$$\begin{aligned} D_{n+1}(c,\omega) = D_n(c,\omega) + \sum_{k=m}^{n-1} \Bigg\{ D_k^*(\omega) \sum_{\ell=0}^{\omega-1} D_{n-k-1}(c-1,\ell) \\ + D_{n-k-1}(c-1,\omega) \sum_{\ell=0}^{\omega-1} D_k^*(\ell) + \\ D_k^*(\omega) D_{n-k-1}(c-1,\omega) \Bigg\} \\ D_n(0,0) = 1,\, D_n(0,d) = D_n(c,0) = 0 \qquad n \leq m+1 \end{aligned} \tag{22}$$

since a structure with a base pair $(1, k+2)$ has order d and c components iff either the bracketed part has order ω and the tail has a order at most ω and $c-1$ components or the bracketed part has a degree smaller than ω and the

tail has $c-1$ components and order ω. It remain to calculate $D_n^*(\omega)$. By inspection we find for $n > m$

$$\begin{aligned} D_n^*(0) &= 0 \\ D_n^*(1) &= 1 + D_n(1,1) \\ D_n^*(\omega) &= D_n(1,\omega) + \sum_{\ell=2}^{\infty} D_k(\ell, \omega-1), \quad \omega \geq 2 \end{aligned} \tag{23}$$

while for $n \leq m$ we have $D_n^*(\omega) = 0$. There is no structure of order 0 with a bracket in it; order one is obtained by either bracketing the open structure or by bracketing a structure with a single component and order 1. If the bracketed part has only a single components its order is preserved by adding a terminal bracket. If it consists of more than one components, the addition of the multi-loop increases the order by one.

Summing over the number of components we obtain the number of Structures with given order $\tilde{D}_n(\omega)$. Let us further introduce the number of structure of order at most one, $D'_n(1)$. It is easy to derive the following system of recursions from the above ones:

$$\begin{aligned} \tilde{D}_{n+1}(\omega) &= \tilde{D}_n(\omega) + \sum_{k=m}^{n-1} \left\{ D_k^*(\omega) \sum_{\ell=0}^{\omega-1} \tilde{D}_{n-k-1}(\ell) + \tilde{D}_{n-k-1}(d) \sum_{\ell=0}^{\omega} D_k^*(\ell) \right\} \\ D_k^*(\omega) &= \tilde{D}_k(\omega-1) + D_k(1,\omega) - D_k(1,\omega-1) \qquad n \geq m+2 \\ D_{n+1}(1,\omega) &= D_n(1,\omega) + \sum_{k=m}^{n-1} D_k^*(\omega) \\ \tilde{D}_n(0) &= 1,\ \tilde{D}_n(\omega) = 0 \text{ for } \omega \geq 1, \qquad n \leq m+1 \end{aligned} \tag{24}$$

For the number of structures with a degree at most one we find

$$\begin{aligned} D'_{n+1} &= D'_n + \sum_{k=m}^{n-1} D_k^*(1) D'_{n-k-1} \\ D_{n+1}^*(1) &= \sum_{k=m}^{n} D_k^*(1) \end{aligned} \tag{25}$$

2.4.4. Secondary Structures with Minimum Stack Length

Let $\Psi_n(l)$ be the number of structures with minimal stack length l, and let $\Psi_n^*(l)$ be the number of structures on n digits which have only stacks of length at least l if an additional terminal base pair is attached. Furthermore let $\Psi_n^{**}(l)$ be the number of structures on n digits with all stacks of length at least l for which $(1, n)$ is not a base pair.

These three numbers fulfil for $l > 1$ the coupled recursions

$$\begin{aligned}
\Psi_{n+1}(l) &= \Psi_n(l) + \sum_{k=m+2l-2}^{n-1} \Psi_k^*(l)\Psi_{n-k-1}(l) \\
\Psi_n^*(l) &= \sum_{p=l-1}^{(n-m)/2} \Psi_{n-2p}^{**}(l) \\
\Psi_n^{**}(l) &= \Psi_n(l) - \Psi_{n-2}^*(l) \\
\Psi_n(l) &= \Psi_{n+1}^{**}(l) = 1 \quad n < m + 2l, \\
\Psi_n^*(l) &= 0 \quad m + 2l - 2
\end{aligned} \tag{26}$$

The first recursion is obvious. A structure which has only stacks of length at least l after addition of the terminal base pair must have a terminal stack of length $p \geq l - 1$. The remaining part of the structure must have stacks of length at least l without a terminal base pair. Of course there is no such structure if $n - 2p < m$. For the numbers $\Psi_n^{**}(l)$ we obtain the explicit recursion:

$$\begin{aligned}
\Psi_{n+1}^{**}(l) &= \Psi_n(l) + \sum_{k=m+2l-2}^{n-2} \Psi_k^*(l)\Psi_{n-k-1}(l) \\
\Psi_n^{**} &= 1 \quad n < m + 2l
\end{aligned} \tag{27}$$

because structures without a terminal base pair and stacks of length at least l are obtained by adding a new base pair to structures which including this base pair have stacks of sufficient length (first factor in the sum) provided the structures in the remaining part of the structure have also sufficient stack length. Of course there may not by a terminal base pair by construction. Comparing the sum in (27) and in the recursion for $\Psi_n(l)$ yields the final result. We have of course $\Psi_n(1) = S_n$ for all n and $\Psi_n(l+1) < \Psi_n(l)$ for all l and sufficiently large n.

Remark. It is possible of course to obtain recursions of the above type for the number of structure elements or the number of structures with particular properties also for $l > 1$. If Ξ_n is the counting series of interest one has to replace $S_k \Xi_{n-k-1}$ by $\Psi_k^* \Xi_{n-k-1}$ and $\Xi_k S_{n-k-1}$ by $\Xi_k^* \Psi_{n-k-1}$, where Ξ^* counts the objects of interest subject to the restriction that the secondary structure has a terminal stack of length at least l.

2.5. Asymptotics

Notation 5.1. The symbols $\sim$ and $\mathcal{O}$ have their usual meaning:
$f(x) = \mathcal{O}(x)$ means $f(x)$ is bounded as $x \to 0$.
$f(n) \sim g(n)$ means $f(n)/g(n) \to 1$ as $n \to \infty$.

The symbol o, however, does not have its standard meaning in this chapter (*cf.* theorem 5.5). If not explicitly stated, asymptotic formulas assume $n \to \infty$.

2.5.1. Asymptotics from Generating Functions

We will use the following simplified version of Darboux' theorem (*cf.* [49, p. 205])

Theorem 5.2. Let $y(x) = \sum_{n=0}^{\infty} y_n x^n$ be of the form

$$y(x) = \beta(x) + \sum_k g_k(x)(1 - \frac{x}{\alpha})^{\omega_k} \tag{28}$$

where β, g_k are analytic on a circle larger than the circle of convergence of $y(x)$, ω_k real but not a non-negative integer. Suppose y has only a single singularity at $x = \alpha$ Denote by ω the smallest exponent ω_k and by $g(x)$ the corresponding analytic factor. Then

$$y_n \sim \frac{g(\alpha)}{\Gamma(-\omega)} n^{-1-\omega} \left(\frac{1}{\alpha}\right)^n \tag{29}$$

Theorem 5.3. [50, Theorem 5]. Assume that $y_n \geq 0$, for n sufficiently larger $y_n > 0$, and $y(x) = \sum_{n=0}^{\infty} y_n x^n$ satisfies $F(x, y) \equiv 0$. Suppose there are real numbers $\alpha > 0$, $\beta > y_0$ such that

(1) for some $\delta > 0$, $F(x, y)$ is analytic whenever $|x| < \alpha+\delta$ and $|y| < \beta+\delta$;
(2) $F(\alpha, \beta) = 0$, $F_y(\alpha, \beta) = 0$;
(3) $F_x(\alpha, \beta) \neq 0$, $F_{yy}(\alpha, \beta) \neq 0$;
(4) (α, β) is the only solution in the interior of the complex rectangle $|x| < \alpha$ and $|y| < \beta$.

Then

$$y_n \sim \sqrt{\frac{\alpha F_x(\alpha, \beta)}{2\pi F_{yy}(\alpha, \beta)}} n^{-3/2} \left(\frac{1}{\alpha}\right)^n \tag{30}$$

Remark 5.4. By comparison of theorem 5.2 and 5.3 we find immediately

$$g(\alpha) = -\sqrt{\frac{2\alpha F_x(\alpha, \beta)}{F_{yy}(\alpha, \beta)}} \qquad \text{with} \quad \beta = \beta(\alpha) \tag{31}$$

Theorem 5.5. Let $\Phi(x, y)$ be analytic for $|x| < \alpha + \delta$ and $|y| < \beta(\alpha) + \delta$, $\delta > 0$. Suppose y is of the form

$$y(x) = \beta(x) + (1 - \frac{x}{\alpha})^{1/2} g(x). \tag{32}$$

Let $z(x) = \sum_{n=0}^{\infty} z_n x^n$ be a generating function of the form $z = \Phi(x, y)$. Then

$$\lim_{n\to\infty} \frac{z_n}{y_n} = \Phi_y(\alpha, \beta(\alpha)) \tag{33}$$

Proof. We will use the short hand o for any analytic function $o(x)$ such that $o(\alpha) = 0$.

$$\begin{aligned} \Phi(x, y) &= \sum_{k=0}^{\infty} a_k(x) y^k = \\ &= \sum_{k=0}^{\infty} a_k(x) \left\{ \beta(x)^k + \left[k\beta(x)^{k-1} g(x) + o \right] (1 - x/\alpha)^{1/2} \right\} = \\ &= \Phi(x, \beta(x)) + \left[\Phi_y(x, \beta(x)) g(x) + o \right] \left(1 - \frac{x}{\alpha}\right)^{1/2} \end{aligned} \tag{34}$$

Darboux' theorem shows that

$$z_n \sim \frac{g(\alpha)\Phi_y(\alpha, \beta)}{\Gamma(-\frac{1}{2})} \cdot n^{-3/2} \left(\frac{1}{\alpha}\right)^n \tag{35}$$

Theorem 5.6. Let $\Phi(x, y)$ be analytic as in the previous theorem. Suppose y is of the form

$$y(x) = \beta(x) + (1 - \frac{x}{\alpha})^{1/2} g(x) \tag{36}$$

Let $z(x) = \sum_{n=0}^{\infty} z_n x^n$ be a generating function of the form

$$z(x) = \frac{1}{\alpha\beta - xy} \Phi(x, y) \tag{37}$$

Then

$$\frac{z_k}{y_k} = \frac{2\Phi(\alpha, \beta)}{\alpha g^2(\alpha)} \cdot n \tag{38}$$

Proof. Consider first

$$\begin{aligned} \frac{1}{\alpha\beta - xy} &= \frac{1}{\alpha\beta - x\beta(x) - xg(x)(1 - x/\alpha)^{1/2}} = \\ &= \frac{\alpha\beta - x\beta(x) + xg(x)(1 - x/\alpha)^{1/2}}{[\alpha\beta - x\beta(x)]^2 - x^2 g^2(x)(1 - x/\alpha)} = \\ &= \frac{o + xg(x)(1 - x/\alpha)^{1/2}}{\mathcal{O}(1 - \frac{x}{\alpha})^2 - x^2 g^2(x)(1 - x/\alpha)} = \\ &= \eta(x) - \frac{1}{xg(x)}[1 + o](1 - x/\alpha)^{-1/2} \end{aligned} \tag{38}$$

where $\eta(x)$ is analytic on circle larger than the circle of convergence of $y(x)$. Multiplying this expression by a Taylor expansion of $\Phi(x, y)$ yields

$$\begin{aligned} \frac{1}{\alpha\beta - xy}\Phi(x, y) &= \eta\Phi - \sum_{k=0}^{\infty} a_k(x)\beta(x)^k \frac{1}{xg(x)}(1 - x/\alpha)^{-1/2}[1 - o] \\ &= \eta\Phi - \frac{\Phi(x, y)}{xg(x)}[1 - o](1 - \frac{x}{\alpha})^{-1/2} \end{aligned} \tag{39}$$

Applying Darboux' theorem and using that $\Gamma(\frac{1}{2}) = -\frac{1}{2}\Gamma(-\frac{1}{2})$ completes the proof.

Corollary 5.7. Let y as in the previous theorem and let u, v be of the same form as z above. Suppose there is an analytic function $\Phi(x, y)$ such that $u = \Phi(x, y)v$. Then

$$\lim_{n\to\infty} \frac{u_n}{v_n} = \Phi(\alpha, \beta) \tag{40}$$

2.5.2. The Number of Secondary Structures

The series S_n has been extensively studied in [36]. Consider the series Ψ_n of secondary structures with a prescribed minimum stack length l and minimum size m for hairpin loops. Denote by

$$\psi(x) = \sum_{n=0}^{\infty} \Psi_n x^n, \qquad \phi(x) = \sum_{n=0}^{\infty} \Psi_n^* x^n, \qquad \theta(x) = \sum_{n=0}^{\infty} \Psi_n^{**} x^n \tag{41}$$

the generating functions. We introduce furthermore the notation

$$t_m(x) = \sum_{k=0}^{m-1} x^k \qquad\qquad \tau_m(x) = \sum_{k=1}^{m-1} k x^k = x\frac{d}{dt} t_m(x) \tag{42}$$

Theorem 5.8. The generating function ψ, ϕ and θ fulfil the coupled functional equations

$$\begin{aligned} \psi &= 1 + x\psi + x^2\phi\psi \\ \phi &= \frac{x^{2(l-1)}}{1-x^2}\big(\theta - t_m(x)\big) \\ \theta &= \psi - x^2\phi \end{aligned} \tag{43}$$

Proof. The first and third line are obvious. The second line yields

$$\begin{aligned} \phi &= \sum_{n=0}^{\infty} x^n \sum_{p=l-1}^{(n-m)/2} \Psi_{n-2p}^{**} = \\ &= \sum_{n=0}^{\infty}\sum_{p=0}^{n/2} x^2 p \Psi_{n-2p}^{**} x^{n-2p} - \sum_{p=0}^{l-2} x^{2p} \sum_{n=0}^{\infty} \Psi_{n-2p}^{**} x^{n-2p} - \\ &\quad - \sum_{p>\frac{n-m}{2}}^{n/2} x^{2p} \sum_{n=0}^{\infty} \Psi_{n-2p}^{**} x^{n-2p} + \sum_{p>\frac{n-m}{2}}^{l-2} x^{2p} \sum_{n=0}^{\infty} \Psi_{n-2p}^{**} x^{n-2p} \\ &= \frac{1}{1-x^2}\theta - \sum_{p=0}^{l-2} x^{2p}\theta - t_m(x) + \sum_{p=0}^{l-2} x^{2p} t_m(x) = \\ &= \frac{x^{2(l-1)}}{1-x^2}\big(\theta - t_m(x)\big) \end{aligned} \tag{44}$$

Corollary 5.9. The generating function ψ fulfils the functional equation

$$F(x,\psi) = x^{2l}\psi^2 - \big[(1-x)(1-x^2+x^{2l}) + x^{2l} t_m(x)\big]\psi + (1-x^2+x^{2l}) = 0 \tag{45}$$

Corollary 5.10. For $l = 1$ we recover the generating function $s(x) = \sum_{n=0}^{\infty} S_k x^k$ for the number of secondary structures. It fulfils the functional equation

$$F(x,y) = 1 + \big(2x - \sum_{k=0}^{m+1} x^k\big)y + x^2 \cdot y^2 = 0 \tag{46}$$

Corollary 5.11.

$$\Psi_n \sim \frac{-g(\alpha)}{2\sqrt{\pi}} n^{-3/2} \left(\frac{1}{\alpha}\right)^n \tag{47}$$

where $g(\alpha)$ is to be taken from equ. (31) using

$$\beta = \frac{1}{\alpha^l}\sqrt{1-\alpha^2+\alpha^{2l}}, \tag{48}$$

and α is the smallest positive solution of

$$2x^l\sqrt{1-x^2+x^{2l}} - \big[(1-x)(1-x^2+x^{2l}) + x^{2l}t_m(x)\big] = 0 \tag{49}$$

Proof. From (43) some simple algebra yield the functional equation (45). From $F(\alpha,\beta) - \beta F_y(\alpha,\beta) = 0$ one obtains immediately (48) and (49) for the zeros necessary for the application of theorem 5.3. The latter proves equ.(47).

Corollary 5.12. For $l = 1$ the above equations simplify to $\beta = 1/\alpha$ and

$$\sum_{k=0}^{m+1} \alpha^k - 4\alpha = 0 \tag{50}$$

Numerical values are given in table 10.

Throughout the remainder of this chapter we will assume $l = 1$ if l is not mentioned explicitly, while α and β will denote the solutions of equations (49) and (48) respectively.

Table 2. Coefficients for the asymptotics of $\Psi_n^{(l,m)}$.

l	$m=0$	1	2	3	5	∞
	α					
1	0.3333	0.3820	0.4142	0.4369	0.4658	0.5000
2	0.4836	0.5081	0.5266	0.5409	0.5610	0.5958
3	0.5672	0.5828	0.5952	0.6053	0.6204	0.6537
4	0.6227	0.6336	0.6428	0.6504	0.6623	0.6938
5	0.6629	0.6712	0.6783	0.6843	0.6941	0.7237
10	0.7704	0.7737	0.7766	0.7793	0.7840	0.8066
20	0.8713	0.8518	0.8530	0.8540	0.8559	0.8713
100	0.9520	0.9521	0.9522	0.9523	0.9525	0.9571
∞	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	$-g(\alpha)/(2\sqrt{\pi})$					
1	1.4658	1.1043	0.8766	0.7131	0.4848	0.0000
2	2.7155	2.1614	1.7742	1.4848	1.0769	0.0000
3	3.9640	3.2711	2.7558	2.3561	1.7741	0.0000
4	5.2305	4.4238	3.7990	3.3003	2.5537	0.0000
5	6.5194	5.6142	4.8923	4.3033	3.4009	0.0000
10	13.309	12.026	10.921	9.962	8.382	0.0000
20	28.365	26.557	24.913	23.414	20.787	0.0000
100	189.31	185.30	181.41	177.63	170.40	0.0000

2.5.3. Average Number of Structure Elements

Denote by Ξ_n the number of structural elements. From the biological point of view it is very interesting to know the average number of structural elements in a single structure, i.e. the asymptotic behavior of Ξ_n/S_n. It is clear that the counting series for the total number of structure elements, including the total number of base pairs and unpaired digits is bounded from above by nS_n.

Lemma 5.13.

$$\begin{aligned} t_m(\alpha) &= \frac{3\alpha-1}{\alpha^2} \qquad \tau_m(\alpha) = \frac{3\alpha-1}{\alpha(1-\alpha)} - m\frac{(1-2\alpha)^2}{\alpha^2(1-\alpha)} \\ g^2(\alpha) &= \frac{(1-2\alpha)(2+m-2m\alpha)}{(1-\alpha)\alpha^3} \end{aligned} \tag{51}$$

Theorem 5.14. For the number of components holds

$$\lim_{n\to\infty} \frac{I_n}{S_n} = 2\beta(1-\alpha) - \beta = 2/\alpha - 3 \tag{52}$$

Proof. Let $i(x) = \sum_{k=0}^{\infty} I_k x^k$ be the generating function for the number of components. The recursion can be brought to the form

$$I_{n+1} = I_n + \sum_{k=0}^{n-1} S_k I_{n-k-1} + \sum_{k=0}^{n-1} S_k S_{n-k-1} - \sum_{k=0}^{m-1} [I_{n-k-1} + S_{n-k-1}] \tag{53}$$

Multiplying by x^{n+1} and summing over n yields

$$i(x) = xi(x) + x^2 s(x) i(x) + x^2 s^2(x) - x^2 t_m(x)[s(x) + i(x)] \tag{54}$$

We find by using twice the functional equation for $s(x)$

$$\begin{aligned} i(x) &= \frac{x^2 s^2(x) - s(x) x^2 t_m(x)}{1 - x - x^2 s(x) + t_m(x)} \\ &= s(x) \cdot x^2 s(x) \big[s(x) - t_m(x) \big] \\ &= s^2(x)(1-x) - s(x) \end{aligned} \tag{55}$$

Application of theorem 5.5 immediately yields the desired result.

Remark 5.15. This result holds for arbitrary minimal stack length l as well.

Theorem 5.16. For the number of external digits holds

$$\lim_{n\to\infty} \frac{E_n}{S_n} = 2\alpha\beta = 2 \tag{56}$$

Proof. The functional equation for the generating function reads $e(x) = x \cdot s^2(x)$. Theorem 5.5 completes the proof.

Theorem 5.17 For the number of unpaired digits holds

$$\frac{U_n}{S_n} \sim \frac{2\alpha + m(1-2\alpha)}{2 + m(1-2\alpha)} \cdot n \tag{57}$$

Proof. Let $u(x) = \sum_{n=0}^{\infty} U_n x^n$ be the generating function of the number of unpaired digits. From recursion (9) we find immediately the functional equation

$$u = xu + xs + 2x^2us - x^2ut_m(x) - x^2s\tau_m(x) \tag{58}$$

Using the functional equation for s, some algebra yields

$$u(x) = \frac{1}{1-x^2s^2} \cdot s^2x(1 - x\tau_m) \tag{59}$$

Application of theorem 5.6 completes the proof.

Remark 5.18. Let $p(x)$ be the generating function of the number of base pairs. Since $U_n + 2P_n = nS_n$ we have $u(x) + 2p(x) = xs'(x)$.

Theorem 5.19. For the number of stacks or loops holds

$$\frac{N_n}{S_n} \sim \frac{(1-\alpha)^2(1+\alpha)}{2 + m - 2m\alpha} \cdot n \tag{60}$$

Proof. Let $\nu(x) = \sum_{n=0}^{\infty} N_n x^n$ be the generating function of the number of stacks. Observe that

$$\sum_{k=m+p}^{n-1} S_{k-p}S_{n-k-1} = \sum_{k=m}^{n-p-1} S_k S_{n-p-k-1} \tag{61}$$

and therefore gives rise to a term $x^{p+2}[s^2 - st_m(x)] = x^p[(1-x)s - 1]$ in the functional equation for the generating function. Thus recursion (6) translates to

$$\nu = x\nu + 2x^2s\nu - x^2\nu t_m(x) + (1-x^2)[s \cdot (1-x) - 1] \tag{62}$$

or, after some simple algebra,

$$\nu = \frac{1}{1-x^2s^2} s(1-x^2)[s(1-x) - 1] \tag{63}$$

The proof is completed by theorem 5.6.

2.5.4. The Number of Structures with Certain Properties

Theorem 5.20. For the number of structures with b base pairs holds

$$H_n(b) \sim \frac{1}{(b+1)!b!} n^{2b} \tag{64}$$

Proof. From recursion (4) one finds the functional equation

$$\begin{aligned} h_b &= x h_b + x^2 \sum_{k=0}^{b-1} h_{b-k-1} h_k - x^2 t_m(x) h_{b-1} \qquad b > 0 \\ &= x h_b + x^2 \sum_{k=1}^{b} h_k h_{b-k-1} + x^{m+2} h_{b-1} \end{aligned} \tag{65}$$

and $h_0(x) = 1/(1-x)$. With the ansatz

$$h_b(x) = \eta_b(x) \cdot \frac{1}{1-x} \left(\frac{x}{1-x} \right)^{2b} \tag{66}$$

one checks finds that $\eta_b(x)$ are polynomials fulfilling

$$\eta_b(x) = \sum_{k=1}^{b} \eta_k(x) \eta b - k - 1(x) + x^m \eta_{b-1} \tag{67}$$

Theorem 5.2 assures now that

$$H_n(b) \sim \frac{\eta_b(1)}{\Gamma(2b+1)} \cdot n^{2b}. \tag{68}$$

Since $\eta_0(1) = 1$, equ.(67) becomes the well known recursion for the Catalan numbers

$$\eta_b(1) = C_b = \frac{1}{b+1} \binom{2b}{b}. \tag{69}$$

Theorem 5.21 For the number of structures with exactly b stacks holds

$$N_n(b) \sim \frac{C_b}{2^b(3b)!} \cdot n^{3b} \tag{70}$$

Proof. Let $\nu_b(x) = \sum_{n=0}^{\infty} N_n(b)x^n$ be the generating function for the number of structures with exactly b stacks and denote by $\zeta_b(x)$ the generating function for the auxiliary variable $Z_n(b)$. It is straight forward to derive the functional equations

$$\begin{aligned} \zeta_b &= \frac{x^2}{(1-x)(1+x)}[\nu_{n-1} - \eta_{b-1}] \\ \nu_b &= \frac{x^2}{(1-x)} \sum_{l=1}^{b} \zeta_l - \nu_{b-l} \end{aligned} \tag{71}$$

One easily checks that these generating functions are of the form

$$\begin{aligned} \nu_b(x) &= \mu_b(x)\frac{1}{(x+1)^b}\frac{1}{(x-1)^{3b+1}} \\ \zeta_b(x) &= \xi_b(x)\frac{1}{(x+1)^b}\frac{1}{(x-1)^{3b+1}} \end{aligned} \tag{72}$$

where $\mu_b(x)$ and $\xi_b(x)$ are polynomials. Theorem 5.2 thus yields

$$N_n(b) \sim \frac{1}{2^b}\frac{\mu_b(1)}{\Gamma(3b+1)} \cdot n^{3b} \tag{73}$$

where $\mu_b(1)$ and $\xi_b(1)$ fulfil the recursions

$$\xi_b(1) = \mu_{b-1}(1) \qquad \mu_n(1) = \sum_{l=1}^{b} \xi_l(1)\mu_{b-l}(1) = \sum_{l=0}^{b-1} \mu_l(1)\mu_{b-l-1}(1) \tag{74}$$

Again, the coefficients $\mu_b(1)$ coincide with the Catalan numbers.

Theorem 5.22. For the number of structures with b hairpins holds

$$A_n(b) \sim \frac{4}{2^{(3+m)b}b!(b-1)!} n^{2(b-1)}2^n \tag{75}$$

Proof. Let $a_b(x)$ denote the generating function $\sum A_n(b)x^n$. From recursion (8) we obtain with some simple algebra

$$a_b = xa_b + x^2 \sum_{i=1}^{b} a_i a_{b-i} + x^2 t_m a_{b-1} \qquad b > 0 \tag{76}$$

and $a_0(x) = 1/(1-x)$. Collecting all terms containing $a_b(x)$ yields

$$a_b(1-2x) = x^{m+2} a_{b-1} + x^2 \sum_{i=1}^{b-1} a_i a_{b-i} \tag{77}$$

With the ansatz

$$a_b(x) = \left(\frac{x^{m+2}}{1-x}\right)^b \frac{1}{(1-2x)^{2b-1}} \eta_b(x) \tag{78}$$

we find the following recursion for the polynomials $\eta_b(x)$:

$$\eta_b(x) = (1-2x)(1-x)\eta_{b-1} + x^2 \sum_{i=1}^{b-1} \eta_i(x)\eta_{b-1} \qquad \eta_1(x) = 1 \tag{79}$$

Theorem 5.2 now implies that the relevant singularity occurs at $x = 1/2$ leaving us with the recursion

$$\eta_b(\tfrac{1}{2}) = \frac{1}{4} \sum_{i=1}^{b-1} \eta_i(\tfrac{1}{2})\eta_{b-i}(\tfrac{1}{2}) \tag{80}$$

It is easy to check that this is solved by

$$\eta_b(\tfrac{1}{2}) = \frac{1}{2^{2(b-1)}} C_{b-1} \tag{81}$$

From theorem 5.2 we find now that

$$A_n(b) \sim \frac{C_{b-1}}{2^{2(b-1)} 2^{b(m+1)} \Gamma(2b+1)} n^{2(b-1)} 2^n \tag{82}$$

Some simple algebra completes the proof.

Theorem 5.23. For the number of structures with b components holds

$$\lim_{n\to\infty} J_n(b)/S_n = \frac{\alpha^2}{(1-\alpha)^3} b \left(\frac{1-2\alpha}{1-\alpha}\right)^{b-1} \tag{83}$$

Proof. Let $j_b(x) = \sum_{n=0}^{\infty} J_n(b)x^n$ be the generating function for the number of secondary structures with exactly b components. Its is straight forward to derive

$$j_b(x) = \frac{x^2}{1-x}\left(s - t_m(x)\right)^b \cdot j_0(x) \qquad b \geq 1 \tag{84}$$

and from $J_n(0) = 1$ we obtain $j_0(x) = 1/(1-x)$. From theorem 5.5 we find that

$$\lim_{n\to\infty} J_n(b)/S_n = \frac{1}{1-\alpha}\left(\frac{\alpha^2}{1-\alpha}\right)^b \cdot b\left(\beta - t_m(\alpha)\right)^{b-1} \tag{85}$$

Theorem 5.24. For the number of structures with b external digits holds

$$\lim_{n\to\infty} E_n(b)/S_n = \frac{1}{4}(b+1)\left(\frac{1}{2}\right)^b \tag{86}$$

Proof. Let $e_b(x)$ be the generating function of the number of secondary structures with exactly b external digits. Recursion (21) yields the functional equation

$$e_b - \delta_{0b} = xe_{b-1} + x^2 s e_b - x^2 e_b t_m(x) \tag{87}$$

Substituting the functional equation for s and some algebra finally yields $e_0 = s/(1+xs)$ and $e_b = \frac{xs}{1+xs}e_{b-1}$. Therefore,

$$e_b = \left(\frac{xs}{1+xs}\right)^b \cdot \frac{s}{1+xs} \tag{88}$$

Application of theorem 5.5 and observing $\alpha\beta = 1$ yields the desired expression.

Theorem 5.25 For any finite order ω there is a positive constant ϵ such that

$$\lim_{n\to\infty} \frac{\tilde{D}_n(\omega-1)e^{\epsilon n}}{\tilde{D}_n(\omega)} = 0 \tag{89}$$

Proof. We will need the generating functions

$$\Delta_\omega = \sum_{n=0}^{\infty} \tilde{D}_n(\omega)x^n \quad \Delta^*_\omega = \sum_{n=0}^{\infty} D^*_n(\omega)x^n \quad \Delta'_\omega = \sum_{n=0}^{\infty} D_n(1,\omega)x^n \tag{90}$$

Recursion (24) yields the following system of coupled functional equations for the above generating functions

$$\begin{aligned} \Delta_\omega &= x\Delta_\omega + x^2\Delta^*_\omega \sum_{i=0}^{\omega-1} \Delta_i + x^2\Delta_\omega \sum_{i=0}^{\omega} \Delta^*_i \\ \Delta^*_\omega &= \Delta_{\omega-1} + \Delta'_\omega - \Delta'_{\omega-1} \qquad \omega \geq 2 \\ \Delta'_\omega &= x\Delta'_\omega + x^2\Delta^*_\omega \frac{1}{1-x} \end{aligned} \tag{91}$$

For $\omega = 0$ we have $\Delta_0 = 1/(1-x)$ and for $\omega = 1$ we find explicitly

$$\begin{aligned} \Delta^*_1(x) &= \frac{1-x}{1-2x}x^m \\ \Delta_1(x) &= \frac{x^{m+2}}{1-x} \cdot \frac{1}{1-2x-x^{m+2}} \end{aligned} \tag{92}$$

Eliminating Δ'_ω we find for $\omega \geq 2$

$$\begin{aligned} \Delta^*_\omega &= \frac{(1-x)^2}{1-2x}\Delta_{\omega-1} - \frac{x^2}{1-2x}\Delta^*_{\omega-1} \\ \Delta_\omega &= \frac{x^2\Delta^*_\omega \sum_{i=0}^{\omega-1} \Delta_i}{1-x-x^2\sum_{i=0}^{\omega} \Delta^*_i} \end{aligned} \tag{93}$$

Unfortunately these expressions become to clumsy to be of much practical use.

Denote $f_\omega(x) = 1 - x - x^2 \sum_{i=0}^{\omega} \Delta^*_i$ and let λ be the unique solution of $1 - 2x - x^{m+2}$ in the interval $[0, 1/2[$. Obviously $f_\omega(x)$ is strictly monotone decreasing and has at least one zero in $(0, \alpha^*)$, where α^* denotes the position of the singularity with the smallest x value among the function $\Delta_i(x)$, $i < \omega$. Therefore, $\Delta_\omega(x)$ has a singularity $\alpha_\omega < \alpha^*$. By induction, therefore, $\alpha_\omega < \alpha_{\omega-1}$ for all ω, since explicitly we have $\alpha_1 = \lambda$ and the first singularity in Δ^*_ω occurs at $x = \alpha_{\omega-1}$. By theorem 5.2 we have $\Delta_n(\omega) \sim c_1 n^{c_2} \alpha_\omega^n$. The inequality $1/\alpha_\omega > 1/\alpha_{\omega-1}$ completes the proof.

Numerical results for the constants obtained for calculating $\Delta_\omega(x)$ explicitly by using MATHEMATICA and numerically solving for the smallest zero of the denominator in (93,2) are tabulated in table 10. The case $m = 1$, $\omega = 1$ has been calculated by Waterman [36, 51].

Table 3. Secondary structures with order ω. The base of the exponential part of the asymptotic is given.

	α_ω		
ω	$m = 0$	$m = 1$	$m = 3$
0	1	1	1
1	0.41421256	0.4533977	0.4863890
2	0.37597060	0.4221456	0.4680050
3	0.35978154	0.4076474	0.4577424

2.5.5. The Distribution of Structure Elements

Theorem 5.26. For the number of loops with b unpaired digits holds

$$\lim_{n\to\infty} \frac{Q_n(b)}{N_n} = \frac{\alpha^2}{(1-\alpha^2)(1-2\alpha)} \times \left[\frac{1}{2\alpha \cdot 2^b} - \Theta(m-b)\alpha^b - (1-2\alpha)\delta_{b0}\right] \tag{94}$$

Proof. Let $q_b(x) = \sum_{k=0}^{\infty} Q_n(b)x^n$ denote the generating function for the number of loops with b unpaired digits. From recursion (13) we find immediately

$$\begin{aligned} q_b &= xq_b + 2x^2sq_b + x^2se_b - x^2q_bt_m - \Theta(m-b)x^b \\ q_0 &= xq_0 + 2x^2sq_0 + x^2se_0 - x^2q_bt_m - \Theta(m) - x^2[s(1-x)-1] \end{aligned} \tag{95}$$

where $\Theta(n)$ denote the Heaviside function, $\Theta(n) = 1$ for $n > 0$ and $\Theta(n) = 0$ for $n \le 0$. Some simple algebra confirms

$$\begin{aligned} q_b &= \frac{1}{1-x^2s^2}x^2s^2[e_b - \Theta(m-b)x^b] \qquad b > 0 \\ q_0 &= \frac{1}{1-x^2s^2}x^2s[se_b - s(1-x) + 1 - \Theta(m-0)] \end{aligned} \tag{96}$$

Corollary 5.7 now proves the assertion.

Theorem 5.27. The asymptotic distribution of stack length is exponential:

$$\lim_{n\to\infty} \frac{W_n(b)}{N_n} = \frac{1-\alpha^2}{\alpha^2}\alpha^{2b} \tag{97}$$

Proof. Let $w_b(x) = \sum_{k=0}^{\infty} W_n(b)x^n$ denote the generating function for the number of stacks of length b. From recursion (16) we find

$$w_b = xw_b + 2x^2sw_b - x^2w_bt_m + (x^{2b+2} - 2x^{2b} + x^{2b-2})[(1-x)s - 1] \tag{98}$$

Some simple algebra assure that

$$w_b = \frac{1}{1-x^2s^2}x^{2b-2}s(1-x^2)^2[(1-x)s-1] = x^{2b-2}(1-x^2)\cdot\nu(x) \tag{99}$$

Theorem 5.6 completes the proof.

2.5.6. Loop Types

Theorem 5.28. The distribution of loop degrees fulfils

$$\begin{aligned}\lim_{n\to\infty}\frac{L_n(d)}{N_n} &= \frac{\alpha^2}{(1-\alpha^2)(1-2\alpha)}\times \\ &\times\left[\frac{1}{1-2\alpha}\left(\frac{1-2\alpha}{1-\alpha}\right)^d - \begin{cases} \frac{3\alpha-1}{\alpha^2} & d=1 \\ (1-2\alpha) & d=2 \\ 0 & d>2 \end{cases}\right]\end{aligned} \tag{100}$$

Proof. Let $\ell_d(x) = \sum_{n=0}^{\infty} L_n(d)x^n$ be the generating function for the number of loops with degree d. For hairpins one finds from recursion (18)

$$\ell_1 = x\ell_1 + 2x^2\ell_1 s - x^2\ell_1 t_m(x) + \frac{x^{m+2}}{1-x}s \tag{101}$$

Similar functional equations can be obtained for loops of higher degree from recursions (19) and (20). They can be brought to the form

$$\begin{aligned}\ell_1 &= \frac{1}{1-x^2s^2}\frac{x^{m+2}}{1-x}s^2 \\ \ell_2 &= \frac{1}{1-x^2s^2}\left[x^2s^2[j_1(x)-(1-x)] + x^2s\right] \\ \ell_d &= \frac{1}{1-x^2s^2}x^2s^2j_{d-1}(x)\end{aligned} \tag{102}$$

Using the explicit expressions for j_d and theorem 5.6, some tedious algebra finally yield equ.(100).

Remark 5.29. The average loop degree $\bar{d}$ can be most easily calculated from the following balance equation which holds for all secondary structures

$$\sum_{\text{loops } \lambda} \deg(\lambda) = 2\#[\text{stacks}] - \#[\text{components}] \tag{103}$$

From equ.(52) and equ.(60) we find immediately that the average loop degree fulfils

$$\lim_{n\to\infty} \bar{d}_n = 2 \tag{104}$$

Theorem 5.30. The ratio of bulges and true interior loops fulfils

$$\lim_{n\to\infty} \frac{B_n}{Y_n} = \frac{2}{\alpha}(1-\alpha) \tag{105}$$

Proof. Denote by $b(x)$ and $y(x)$ the generating function for the number of bulges and interior loops respectively. By construction they fulfil $b(x) + y(x) = \ell_2(x)$. It is thus sufficient to calculate $y(x)$ from recursion (18). We find

$$y(x) = \frac{1}{1-x^2s^2} s^2x^4 j_1(x) \tag{106}$$

and thus

$$b(x) = \ell_2(x) - y(x) = \frac{1}{1-x^2s^2} x^2 s \left[s(1-x^2)j_1 - (1-x)s + 1 \right] \tag{107}$$

Corollary 5.7 completes the proof.

2.6. Secondary Structures of a Sequence

Up to now we have neglected the fact that secondary structures are built on sequences. Not all secondary structures can be formed by a given biological sequence, since not all combinations of nucleotides form base pairs. The results of the previous sections will be generalized to this situation in the following.

Definition 6.1. Let $\mathcal{A}$ be some finite alphabet of size κ, let Π be a symmetric Boolean $\kappa \times \kappa$-matrix and let $\Sigma = [\sigma_1 \dots \sigma_N]$ be a string of length N over $\mathcal{A}$.

A secondary structure is *compatible* with the sequence Σ if for all base pairs (p, q) holds $\Pi_{\sigma_p,\sigma_q} = 1$.

Following [47, 36] the number of secondary structures $\mathcal{S}$ compatible with some string can be enumerated as follows: Denote by $S_{p,q}$ the number of structures compatible with the substring $[\sigma_p \dots \sigma_q]$. Then

$$S_{l,n+1} = S_{l,n} + \sum_{k=l}^{n-m} S_{l,k-1} S_{k+1,n} \Pi_{\sigma_k,\sigma_{n+1}} \tag{108}$$

For a random sequence, the expected number $\bar{S}_n$ of compatible structures is then [41]

$$\bar{S}_{n+1} = \bar{S}_n + p \sum_{k=1}^{n-m} \bar{S}_{k-1} \bar{S}_{n-k} = \bar{S}_n + p \sum_{k=m}^{n-1} \bar{S}_k \bar{S}_{n-k-1} \tag{109}$$

where

$$p = \frac{1}{\kappa^2} \sum_{i,j=1}^{\kappa} \Pi_{ij} \tag{110}$$

is called the *stickiness* [52].

Remark 6.2 A secondary structure compatible with a given sequence with maximal number of base pairs can be determined by a dynamic programming algorithm [53]. This observation was the starting point for the construction of reliable energy-directed folding algorithms (see next chapter).

Analogously one can derive recursions of the number of structure elements compatible with a string Σ. All recursions in this chapter are sums of linear terms of the form A_n and quadratic terms of the type

$$\sum_{k=m}^{n-1} B_k C_{n-k-1} = \sum_{k=1}^{n-m} C_{k-1} B_{n-k} \tag{111}$$

The corresponding recursions for structures compatible with a string can then be found by the rule

$$\begin{aligned} A_n &\longrightarrow A_{l,n} \\ \sum_{k=1}^{n-m} C_{k-1} B_{n-k} &\longrightarrow \sum_{k=l}^{n-m} C_{l,k-1} B_{k+1,n} \Pi_{\sigma_k,\sigma_{n+1}} \end{aligned} \tag{112}$$

For expected numbers for random sequences the above rules simplify to

$$\begin{aligned} A_n &\longrightarrow A_n \\ \sum_{k=m}^{n-1} B_k C_{n-k-1} &\longrightarrow p \sum_{k=m}^{n-1} B_k C_{n-k-1} \end{aligned} \tag{113}$$

As an example we calculate the expected proportion of unpaired digits in a secondary structure compatible with a random sequence with stickiness p. Application of the above rules to equ.(9) immediately yields

$$\begin{aligned} U_{n+1} &= (U_n + S_n) + p \sum_{k=m}^{n-1} [S_k U_{n-k-1} + S_{n-k-1} U_k], \qquad n \geq m+1 \\ U_n &= n, \quad n \leq m+1 \end{aligned} \tag{114}$$

From equ.(109) and equ.(114) we obtain the generating functions

$$1 = s[1 - x - px^2 s + px^2 t_m] \tag{115}$$

$$u = xu + xs + p[2x^2 us - x^2 u t_m - x^2 s \tau_m] \tag{116}$$

For α and β we find from the functional equation for s

$$\begin{aligned} &\alpha\beta = 1/\sqrt{p} \\ &\frac{1}{\sqrt{p}} - (2 + \frac{1}{\sqrt{p}})\alpha + \sqrt{p}\alpha^2 t_m(\alpha) = 0 \end{aligned} \tag{117}$$

Theorem 5.3 allows to calculate the asymptotic for $\bar{S}_n(p)$, yielding the following generalization of lemma 5.13

Lemma 6.3

$$\begin{aligned} t_m(\alpha) &= \frac{(1 + 2\sqrt{p})\alpha - 1}{p\alpha^2} \\ \tau_m(\alpha) &= \frac{(1 + 2\sqrt{p})\alpha^2 - \alpha - m\left(1 - (1 + \sqrt{p})\alpha\right)^2}{p\alpha^2(1 - \alpha)} \\ g^2(\alpha) &= \frac{(1 - \alpha - \sqrt{p}\alpha)(2 + m(1 - \alpha - \sqrt{p}\alpha))}{\sqrt{p}^3(1 - \alpha)\alpha^3} \end{aligned} \tag{118}$$

Table 4. Asymptotics of some structure elements as a function of stickiness

p	1	0.5 **GC**	0.375 **AUGC**	0.25 **GCXK**
α	0.4369	0.5092	0.5391	0.5809
U_n/nS_n	0.5265	0.5897	0.6147	0.6487
P_n/nS_n	0.2368	0.2051	0.1926	0.1756
N_n/nS_n	0.1915	0.1786	0.1717	0.1608
I_n/S_n	1.5776	1.7266	1.7918	1.8855
$L_n(1)/N_n$	0.2769	0.3062	0.3183	0.3352
$L_n(2)/N_n$	0.5082	0.4692	0.4537	0.4325
B_n/Y_n	2.5776	1.9280	1.7096	1.4428
StackLength	1.2363	1.1487	1.1220	1.0924
LoopSize	2.7493	3.3018	3.5801	4.0342
E_n/S_n	2	2.828	3.266	4

Equation (116) thus simplifies to

$$u = \frac{s^2 x\,(1 - p\,\tau_m x)}{1 - p s^2 x^2} \tag{119}$$

and theorem 5.6 implies that

$$\lim_{n\to\infty} \frac{U_n}{nS_n} = \frac{1}{\alpha g^2(\alpha) p}\left[\frac{1}{\sqrt{p}\alpha} - \sqrt{p}\tau_m(\alpha)\right] = \frac{2\alpha + m(1-\alpha-\sqrt{p}\alpha)}{2 + m(1-\alpha-\sqrt{p}\alpha)} \tag{120}$$

The asymptotics of the most important other series are given below without proofs. Numerical values for the most common values of stickiness are given in table 4. The value $p = \frac{1}{2}$ corresponds to a binary alphabet of complementary bases, while $p = \frac{1}{4}$ corresponds to a four letter alphabet with two pairs of complementary bases as in the (such as the biophysical **AUGC** with Watson-Crick pairing rules). Since biological RNA structures frequently contain G-U pairs as well, they are best modeled by a value of $p = \frac{3}{8}$.

Number of Loop and stacks:

$$\begin{aligned} \nu &= \frac{s\,(1 - s(1-x))\,(px^2 - 1)}{1 - p s^2 x^2} \\ \lim_{n\to\infty} \frac{N_n}{S_n} &= \frac{(1-\alpha)(1-\alpha^2 p)}{2 + m(1-\alpha-\alpha\sqrt{p})} \end{aligned} \tag{121}$$

Number of components:

$$\begin{aligned} i &= s^2(1-x) - s \\ \lim_{n\to\infty} \frac{I_n}{S_n} &= 2\beta(1-\alpha) - 1 \end{aligned} \tag{122}$$

Loops with degree 2, i.e., interior loops and bulges:

$$\begin{aligned} l_2 &= \frac{psx^2\left[(1-x)^2 - s(1-x)^3 + psx^2(s-t_m)\right]}{(1-x)^2(1-ps^2x^2)} \\ \lim_{n\to\infty} \frac{L_n(2)}{N_n} &= \frac{(2-\alpha)\alpha^3 p}{(1-\alpha)^2(1-\alpha^2 p)} \\ \lim_{n\to\infty} \frac{B_n}{Y_n} &= 2/\alpha - 2 \end{aligned} \tag{123}$$

Hairpins:

$$\begin{aligned} l_1 &= \frac{ps^2x^2\left(1-(1-x)t_m\right)}{(1-x)\left(1-ps^2x^2\right)} \\ \lim_{n\to\infty} \frac{L_n(1)}{N_n} &= \frac{1-\alpha-\alpha\sqrt{p}}{1-\alpha-\alpha^2 p+\alpha^3 p} \end{aligned} \tag{124}$$

3. RNA Secondary Structure Prediction

3.1. Overview

Several methods exist for the prediction of RNA secondary structures. They can be divided into two broad classes: Folding by phylogenetic comparison and energy directed (i.e. kinetic or thermodynamic) folding.

3.1.1. Comparative Structure Analysis

Given a large enough number of sequences with identical secondary structure, that structure can be deduced by examining covariances of nucleotides in these sequences. This is the principle used for structure prediction through phylogenetic comparison of homologous sequences [54].

The underlying assumption is that structure is more conserved during evolution than sequence, since it is the structure that determines function. In fact the success of the method in the prediction of, for instance, the secondary structures of the 16S ribosomal RNAs [55] provides an excellent justification for this assumption.

Basically these methods just look for compensatory mutations such as an A changing to C in position i of the aligned sequences simultaneously with a change from U to G in position j, indicating a base pair (i, j). The well known clover-leaf structure of tRNAs was found in this manner by just looking at a few sequences. Since no assumption about pairing rules are necessary, non-canonical pairs and tertiary interactions [56] can be detected as well.

Phylogenetic comparison can generate the most reliable secondary structure models to date, provided the set of sequences is sufficiently large, and exhibits the right amount of variation. The sequences should be dissimilar enough to show many covariations while still yielding a good alignment. Such data sets exist in particular for transfer and ribosomal RNAs, the structures thus

determined are therefore frequently used for comparison of other folding algorithms. It should be kept in mind, however, that phylogenetically determined structures usually are incomplete, that is, they do not show all base pairs of the actual structures. This will happen in particular when parts of the sequence are conserved (i.e. the function is sequence dependent) or parts of the structure are variable (because non-functional).

3.1.2. Energy Directed Folding

Most methods for prediction of RNA secondary structure work on the basis of the same energy model that will be presented in the next section. They can be further divided into methods that try to find the structure of minimal energy (or the equilibrium ensemble) and "kinetic" algorithms. It is by no means clear that the biological relevant structure of RNA molecules is the structure of minimal energy, instead the structure might be trapped in some local minimum during the folding process. Kinetic algorithms therefore try to mimic the folding process in order to derive the biologically active structure.

The first kinetical algorithm was proposed by Martinez [57] in 1984, mainly as an attempt to create a *faster* algorithm. As do many other algorithms it starts by compiling a list of possible helices. His idea was that the helix with largest equilibrium constant (that is the lowest energy) would form first. All helices not compatible with this helix are then deleted from the list. The process is repeated until no helix is left whose incorporation would lower the energy of the structure. Such an algorithm will indeed execute in only $\mathcal{O}(n^2)$ steps. The procedure implies that a helix that has once formed never opens again, so there is no re-folding. Furthermore, folding in vivo already starts during transcription, so that helices near the $5'$ end of the sequence should be formed first. Recently interest in this kind of algorithm was renewed [58, 59] because, as in all stem oriented methods, pseudo knots can be easily included. The prediction of pseudo knots, however, remains a problem, since not enough experimental values are available to assign reliable energy values to them.

Currently the only kinetic algorithms allowing for re-folding of the sequence are nondeterministic. The folding of the molecule is typically simulated by

Monte Carlo methods [60]. Such an algorithm can also be modified to mimic $5' \rightarrow 3'$ folding of RNA during transcription[61]. In principle these methods can produce the Boltzmann weighted equilibrium ensemble of structures. They are, however, too time consuming to be usable for sequences much longer than tRNAs.

Manfred Tacker in our group has recently implemented a kinetic folding algorithm [62, 63] similar to Martinez'. However, only helices more stable than some threshold are kinetically determined the rest of the structure is then "filled" up using the minimum free energy algorithm to be described later. A variant for $5' \rightarrow 3'$ is available as well. Comparison of the results of his algorithm with phylogentically determined structures for eight 16S rRNAs showed no significant improvement over the minimum free energy algorithm.

Early algorithms for the calculation of minimum free energy structures, used a list of possible helices and found the optimal structure by enumeration [64] of all possible combinations. By only considering helices of a minimal size, the number of possible combinations can be made small enough for enumeration to be feasible. However, as has been shown in the previous chapter the number structures to consider will always rise exponentially, so that such algorithms can only be practical only for small sequences up to about 200 nucleotides. As it is easy to accommodate pseudo knots and more complicated energy rules, they may be still useful to test new models on small examples. They have been little used since it became clear that minimum free energy structures can be calculated much faster by the kind of dynamic programming algorithms that will be described later.

3.2. The Energy Model

The crucial ingredient for any reliable structure prediction algorithm is a reliable energy model. Luckily, the currently accepted model for RNA secondary structure has a mathematically simple form that allows for an elegant solution of the minimum free energy problem. The energy of structure can be written as a sum of independent contributions for each loop of the structure

$$E(S) = \sum_{\text{loops } L \text{ in } S} e(L) + e(L_{ext}), \tag{125}$$

where L_{ext} is the contribution of the "exterior" loop containing the free ends. Note that here stacked pairs are treated as minimal loops of degree 2.

Empirical energy parameters for calculation of the $e(L)$ have been derived mostly from melting experiments on small oligonucleotides using a nearest-neighbor model. The assumption here is that the energy of some loop only depends only on the size and type of the loop, the pairs closing the loop and the bases directly adjacent to these pairs. The first compilation of such parameters was done by Salser [65]. The parameters most widely in use today are taken from Freier et al. [66, 67], they were measured at 37°C in 1 M NaCl.

In particular the energy model contains the following contributions:

Stacked pairs contribute the major part of the energy stabilizing a structure. Surprisingly the parallel stacking of base pairs is more important than the hydrogen bonding of the complementary bases. By now all 21 possible combinations of AU GC and GU pairs have been measured in several oligonucleotide sequences with an accuracy of a few percent. The parameters involving GU mismatches were measured more recently in Douglas Turner's group [68] and brought the first notable violation of the nearest-neighbor model: while all other combinations could be fitted reasonably well to the model, the energy of the $\begin{smallmatrix}5'\mathrm{GU}\,3'\\3'\mathrm{UG}\,5'\end{smallmatrix}$ stacked pair seems to vary form +1.5kcal/mol to −1.0kcal/mol depending on its context.

Dangling ends: unpaired bases adjacent to a helix may also lower the energy of the structure through parallel stacking. In the case of free ends the bases dangling on the $5'$ and $3'$ ends of the helix are evaluated separately, unpaired nucleotides in multi-loops are treated in the same way. For interior and hairpin loops the so called **terminal mismatch** energy depends on the last pair of the helix and both neighboring unpaired bases. While stacking of an unpaired base at the $3'$ end can be as stabilizing as some stacked pairs, $5'$ dangling ends usually contribute little stability. Terminal mismatch energies are often similar as the sum of the two corresponding dangling ends. Typically, terminal mismatch energies are not assigned to hairpins of size three. Few measurements are available for the stacking of unpaired nucleotides on GU pairs so that they have to be estimated from the data for GC and AU pairs.

Loop energies are destabilizing and modeled as purely entropic. Few experimental data are available for loops, most of these for hairpins. The parameters for loop energies are therefore particularly unreliable. Data in the newer compilation by Jaeger et al. [69] differ widely from the values given previously [66]. Energies depend only on the size and type (hairpin, interior or bulge) of the the loop. Hairpins must have a minimal size of 3, values for large loops are extrapolated logarithmically. Asymmetric interior loops are furthermore penalized [70] using an empirical formula depending on the difference $|u_1 - u_2|$ of unpaired bases on each side of the loop. For bulge loops of size 1 a stacking energy for the stacking of the closing and the interior pair is usually added, while larger loops are assumed to prohibit stacking. Finally, a set of eight hairpin loops of size 4 are given a bonus energy of 2kcal/mol. These tetraloops have been found to be especially frequent in rRNA structures determined from phylogenetic analysis. Melting experiments on several tetraloops were performed recently [71] showing a strong sequence dependence that is not yet well reflected in the energy parameters.

No measured parameters are available for multi-loops, their contribution (apart from dangling ends within the loop) is usually approximated by the linear ansatz

$$\Delta G = a + bu + cm \tag{126}$$

where u is the size of the loop and m is the number of base pairs interior to the loop, i.e. its degree-1. Good results have been achieved using $a = 4.6$, $b = 0.4$ and $c = 0.1$ kcal/mol. While a logarithmic size dependency of loop energies would be more realistic, the linear ansatz allows faster prediction algorithms. Since all energies are measured relative to the unfolded chain free ends do not contribute to the energy.

Numerical values for the energy parameters used in the rest of this work can be found in the appendix.

3.3. Dynamic Programming Folding Algorithms

3.3.1. Calculation of Minimum Free Energy Structures

The additive form of the energy model in equ. (125) allows for an elegant solution of the minimum free energy problem through dynamic programming, similar to sequence alignment. This similarity was first realized and exploited by Waterman [36, 72], the first dynamic programming solution was proposed by Nussinov [53, 73] originally for the "maximum matching" problem of finding the structure with the maximum number of base pairs. The algorithm to be described here is a variant of Zuker and Stiegler's [74, 41].

The algorithm essentially works by calculating optimal structures for all subsequences of the sequence I to be folded, proceeding from smaller to larger fragments. Let C_{ij} be the the minimum energy possible on the substructure I_{ij} given that i and j pair. Since the energy of some substructure $\mathcal{S}_{ij}$ with i and j paired is given by the energy of the loop closed by (i, j) plus the energy of any loops directly interior to it,

$$C_{ij} = \min_{\substack{\text{loops } L \\ \text{closed by } i,j}} \left\{ E(L) + \sum_{\substack{\text{interior pairs} \\ (p,q)\in L}} C_{pq} \right\} \tag{127}$$

and $C_{ii} = \infty$. The unconstrained minimum energy F_{ij} is then given by

$$\begin{aligned} F_{ij} = \min\Big\{ & C_{ij},\ (d_{i;i+1,j} + C_{i+1,j}),\ (C_{i,j-1} + d_{i,j-1;j}), \\ & (d_{i;i+1,j-1} + C_{i+1,j-1} + d_{i+1,j-1;j}),\ \min_{i\le h<j}(F_{ih} + F_{h+1,j}) \Big\} \end{aligned} \tag{128}$$

where the d terms are the contributions from dangling ends and $F_{ii} = 0$. If loops up to a maximal degree of k are considered, evaluation of equ. (128) for every i and j takes time proportional to n^{2k}, it will therefore only be used for loops of degree ≤ 2. The linear ansatz for the energy of multi loops in equ. (126) allows those contributions to be evaluated in $\mathcal{O}(n^3)$ steps. To do this we define another array F^M_{ij} that holds the minimal energy on the subsequence I_{ij} given that i and j are part of a multi loop. Let (i, j) be the

closing base pair of a multi loop, then neglecting dangling ends C_{ij} will be given by

$$C_{ij} = \min_{i<h<j} \left\{ F^M_{i+1,h} + F^M_{h+1,j} \right\} + a. \tag{129}$$

Dangling ends lead to three further terms the equation corresponding to stacking of the unpaired base on the $5'$, $3'$ or both sides of the pair (i,j), such as $d_{i,j;i+1} + F^M_{i+2,h} + F^M_{h+1,j}$ for the case of a $3'$ dangle. The F^M_{ij} can again be calculated recursively in analogy to equ. (128) with the initial condition $F^N_{ii} = \infty$:

$$\begin{aligned} F^M_{ij} = \min\Big\{ & C_{ij} + c,\ (d_{i;i+1,j} + C_{i+1,j} + c),\ (C_{i,j-1} + d_{i,j-1;j} + c), \\ & (d_{i;i+1,j-1} + C_{i+1,j-1} + d_{i+1,j-1;j} + c),\ F^M_{i+1,j} + b,\ F^M_{i,j-1} + b, \\ & \min_{i<h<j-1} (F^M_{ih} + F^M_{h+1,j}) \Big\} \end{aligned} \tag{130}$$

Table 5. Pseudo Code of the minimum free energy folding algorithm.

```
for(d=1...n)
   for(i=1...n-d)
      j=i+d
      C[i,j] = MIN(
          Hairpin(i,j),
          MIN( i<p<q<j : Interior(i,j;p,q)+C[p,q] ),
          MIN( i<k<j : FM[i+1,k]+FM[k+1,j-1]+a )       )
      F[i,j] = MIN( C[i,j], MIN(i<k<j : F[i,k]+F[k+1,j]))
      FM[i,j]= MIN( C[i,j]+c, FM[i+1,j]+b, FM[i,j-1]+b,
                    MIN( i<k<j : FM[i,k]+FM[k+1,j] )       )
free_energy = F[1,n]
```

Remark. `F[i,j]` denotes the minimum energy for the subsequence consisting of bases i through j. `C[i,j]` is the energy given that i and j. pair. The array `FM` is introduced for handling multi-loops. The functions `Interior(i,j;p,q)` and `Hairpin(i,j)` denote the energy contribution of a loop closed by the base pairs given as argument. We have assumed that multi-loops have energy contribution `F=a+b*U+c*I`, where `I` is the number of interior base pairs and `U` is the number of unpaired digits of the loop. The time complexity here is $\mathcal{O}(n^4)$. It is reduced to $\mathcal{O}(n^3)$ by restricting the size of interior loops to some constant, say 30. Dangling ends have been neglected for simplicity.

Evaluation of equ. (127) still gives an $\mathcal{O}(n^4)$ algorithm. Large loops are, however, very destabilizing and therefore rare. As seen in equation 94 their frequency decreases exponentially even for random structures. It is therefore reasonable to restrict the search to interior loops with a maximal loop size, cutting the execution time of that part down to $\mathcal{O}(n^2)$. A maximum loop size of 30 has usually proved sufficient. A compact pseudo-code for the algorithm is given in table 6.

The algorithm outlined so far only calculates minimum energies, not structures. Although it is easy to record for every minimum in the equations above the structural motif that yielded the minimum, storage is often the more stringent constraint on the algorithm than execution time. Typical implementations will therefore first calculate all entries in the F, C and F^M arrays starting with the smallest subsequences and then construct the structure in a second pass proceeding from the exterior loop to the smallest substructures. This technique, typical for dynamic programming, is called backtracking. Since only those entries in the arrays belonging to the minimum energy solution have to be recalculated, the structure can be obtained in less than $\mathcal{O}(n^2)$ steps and negligible time.

3.3.2. Calculation of the Partition Function

The algorithm described above yields only a single structure with minimal free energy. This is unsatisfactory for two reasons: An RNA molecule will in reality not always stay in its minimal energy configuration, but change between many structures of similar energy, possibly in a way related to the molecules function. Secondly, if several structures have energies very close to the minimum, choosing one of them becomes arbitrary because of the inaccuracies of the energy model. To overcome this problem Michael Zuker [75] devised a modification of the minimum free energy algorithm that can efficiently generate all structures within a prescribed increment of the minimum. An even more elegant solution was presented a few years ago by John McCaskill [76] who noticed that the partition function Q of all secondary structures can be calculated by dynamic programming as well. The free energy of the ensemble can be obtained as $F = -kT \ln Q$. While the minimum free energy described before is a free energy only insofar as it contains

those entropy terms inherent in the energy parameters, the free energy of the ensemble additionally includes the entropy stemming from the different structures of the sequence.

Calculation of the partition function resembles closely the minimum free energy algorithm presented before. Generally the partition function is given by

$$Q = \sum_{\text{all structures } S} e^{-\frac{\Delta G(S)}{kT}}. \tag{131}$$

Using the additive form of the energy model (equ. 125) we get

$$Q = \sum_{\substack{\text{all structures} \\ S}} \prod_{\substack{\text{loops} \\ L \in S}} e^{-E(L)/kT}. \tag{132}$$

This allows us again to set up a recursion in close analogy to equ (127) for the partition function Q^b_{ij} of a subsequence I_{ij} given that i and j pair.

$$Q^b_{ij} = \sum_{\text{loops } L\top \text{ closed by } (i,j)} e^{-E(L)/kT} \prod_{\substack{\text{interior pairs} \\ (p,q)\in L}} Q^b_{pq}, \tag{133}$$

from which the partition function can be obtained as

$$Q_{ij} = 1 + \sum_{\substack{h,l \\ i<h<l\leq j}} Q_{i,h-1} Q^b_{hl}. \tag{134}$$

Construction of the algorithm then proceeds as outlined before for the minimum free energy algorithm, using sums instead of minima and products instead of sums. Care must be taken, however, never to count a contribution twice. The algorithm therefore needs more storage than the minimum free energy variant. For details see the pseudo-code in table 6. Also, inclusion of dangling ends is slightly more complicated, so that no implementation using these is available at the moment.

Clearly such an algorithm does not predict a secondary structure, instead

Table 6. Pseudo code for the calculation of the partition function.

```
for(d=1...n)
   for(i=1...n-d)
      j=i+d
      QB[i,j] = EHairpin(i,j) +
         SUM( i<p<q<j : EInterior(i,j;p,q)*QB[p,q] ) +
         SUM( i<k<j : QM[i+1,k-1]*QM1[k,j-1]*Ea )
      QM[i,j] =
         SUM( i<k<j : (Eb^(k-i)+QM[i,k-1])*QM1[k,j] )
      QM1[i,j]= SUM( i<k<=j : QB[i,k]*Eb^(j-k)*Ec )
      Q1[i,j  = SUM( i<l<=j : QB[i,l] )
      Q[i,j]  = 1 + SUM( i<k<=j : Q[i,k-1]*Q1[k,j] )
partition_function = Q[1,n]
```

Remark. Here Ex:=exp$(-x/kT)$ denotes the Boltzmann weights corresponding to the energy contribution x. `Q[i,j]` denotes the partition function Q_{ij} of the subsequence i through j. The array `QM` contains the partition function Q^b_{ij} of the subsequence subject to the fact that i and j form a base pair. `QM` and `QM1` are used for handling the multi-loop contributions, `Q1` is an auxiliary array to reduce the time complexity to $\mathcal{O}(n^3)$. `x^y` means x^y.

one can calculate the probability P_{kl} for the formation of a base pair (k,l):

$$P_{kl} = \frac{Q_{1,k-1}Q^b_{kl}Q_{l+1,n}}{Q} + \sum_{\substack{i,j \\ i<k<l<j}} \frac{P_{ij}Q^b_{kl}\texttt{EInterior}(i,j;k,l)}{Q^b_{ij}} + e^a e^c \sum_{\substack{i,j \\ i<k<l<j}} \frac{P_{ij}}{Q^b_{kl}} \Big[e^{b(k-i-1)} Q^m_{l+1,j-1} + Q^m_{i+1,k-1} e^{b(j-l-1)} + Q^m_{i+1,k-1} Q^m_{l+1,j-1} \Big] \tag{135}$$

Though the number steps necessary for the calculation of the P_{kl} can be reduced to $\mathcal{O}(n^3)$ with the introduction of two auxiliary arrays, the backtracking is here as expensive as the calculation of the Q. In similar ways one could calculate the probabilities of other structural motifs such as certain loop types as well. The heat capacity can be helpful to predict melting temperatures, since structural re-arrangements are accompanied by peaks in the heat capacity.

Heat capacity of RNA secondary structures

Another possible use of the partition function is the calculation of heat capacities, which can be obtained from the well known formula

$$C_p = -T\frac{\partial^2 G}{\partial T^2}, \qquad \text{and} \qquad \Delta G = -kT \ln Q. \tag{136}$$

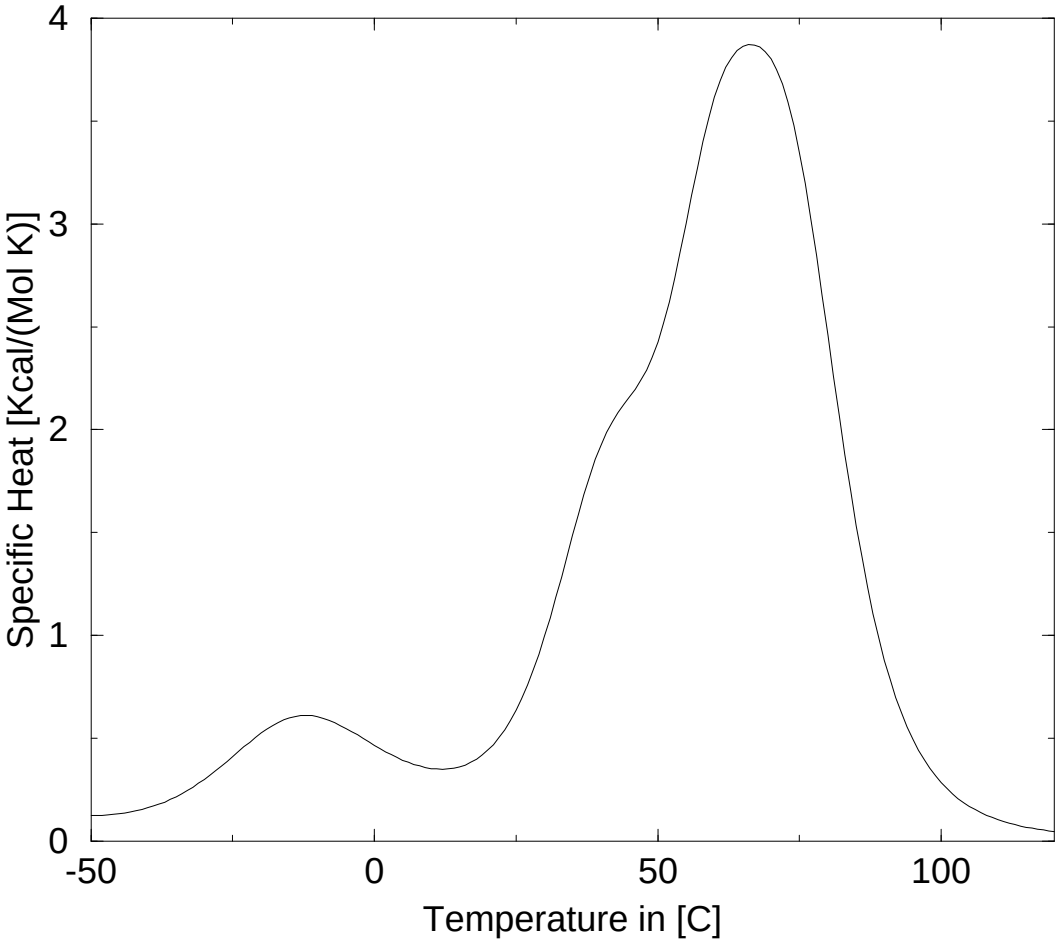


Figure 10: Heat Capacity of the tRNA-phe from yeast in the range $-50°$C to $120°$C.

For the numerical calculation of the heat capacity we have to evaluate the partition function at different temperatures and then perform a numerical differentiation. To obtain a smooth function the numerical differentiation is performed in the following way: We fit the function $F(x)$ by the least square

parabola $y = cx^2 + bx + a$ through the $2m+1$ equally spaced points $x_0 - mh$, $x_0 - (m-1)h$, ..., x_0, ..., $x_0 + mh$. The second derivative of F is then approximated by $F''(x_0) = 2c$. Explicitly, we obtain

$$F''(x_0) = \sum_{k=-m}^{m} \frac{30\left(3k^2 - m(m+1)\right)}{m(4m^2-1)(m+1)(2m+3)} F(x_0 + kh). \tag{137}$$

As an example we show the heat capacity of tRNA^{phe} from yeast (fig. 10). The peak at about $-10°\text{C}$ marks the transition to the usual clover-leaf structure. The slight bump at about $40°\text{C}$ corresponds to the melting of the stem closing the multi loop, while the three hairpins melt only above $60°\text{C}$.

3.4. Inverse Folding

While the problem of predicting RNA secondary structures has been studied extensively for a long time, we are not aware of other work on the inverse problem of finding sequences that will fold into a preselected structure, although the equivalent problem for protein folding has recently received some attention [77]. Such an algorithm could have several uses: Later on we will use it to investigate the distribution of sequences with the same structure in sequence space. Similarly it can be used to estimate the frequency of a given structure. Biotechnological applications can be envisioned as well. Inverse folding may allow to predict novel sequences that are functionally equivalent but unrelated to natural occurring RNAs.

Since the mapping from RNA sequences to structures is a many to one mapping there may be an astronomical number of sequences with a common structure. For that reason alone, it is hard to imagine a deterministic algorithm for the inverse folding problem. The algorithm proposed here is therefore heuristic and treats the task essentially as an optimization problem. It is based on minimum free energy folding but could in principle accommodate other methods as well. Such a search algorithm should ideally have several partly conflicting properties:

- The search should be successful often, even if the desired structure is rare.

- It should find many structures in short time, especially for frequent structures.
- The algorithm should introduce no bias. I.e. the found sequences should be as random as possible.
- The search should be local, to allow us to look for solutions in the vicinity of some reference sequence.

Only compatible sequences are considered as candidates in the inverse folding procedure. Clearly, a compatible sequence can but need not have the target structure as its minimum free energy structure.

Our basic approach is to modify an initial sequence I_0, such as to minimize a cost function given by some distance $f(I) = d(S(I), \mathcal{T})$ between the structure $\mathcal{S}(I)$ of the test sequence I and the target structure $\mathcal{T}$, e.g. a structure distance as described in chapter 2.

Such an procedure may be extremely slow, since it requires many evaluations of the cost function and thereby many executions of the folding algorithm, whose time complexity increases as n^3 with sequence length n. One important way to reduce the time for the computation lies in doing the search recursively. Instead of running the optimization directly on the full length sequence, we optimize small substructures first, proceeding to larger ones as shown in the flow chart (figure 12). This is possible because substructures contribute additively to the energy. If $\mathcal{T}$ is an optimal structure on the sequence $\mathcal{S}$ and contains the base pair (i, j) then the substructure $\mathcal{T}_{i,j}$ must be optimal on the subsequence $\mathcal{S}_{i,j}$, given that i and j pair. Similarly each component of $\mathcal{T}$ must be optimal and for every interior base pair (i, j) of a multi loop $\mathcal{T}_{i,j}$ must be optimal on $I_{i,j}$ given that i, j are part of a multi loop. It is likely then, but by no means necessary, that the converse also holds: A structure that is optimal for a subsequence will also appear with enhanced probability as a substructure of the full sequence. This approach reduces the probability of getting stuck in a local minimum, and more importantly, it reduces the number of foldings of full length sequences, effectively trading foldings of long sequences against (maybe a few more) foldings of much shorter sequences. A full length sequence will not be folded before all necessary conditions deriving form the optimality of substructures are met.

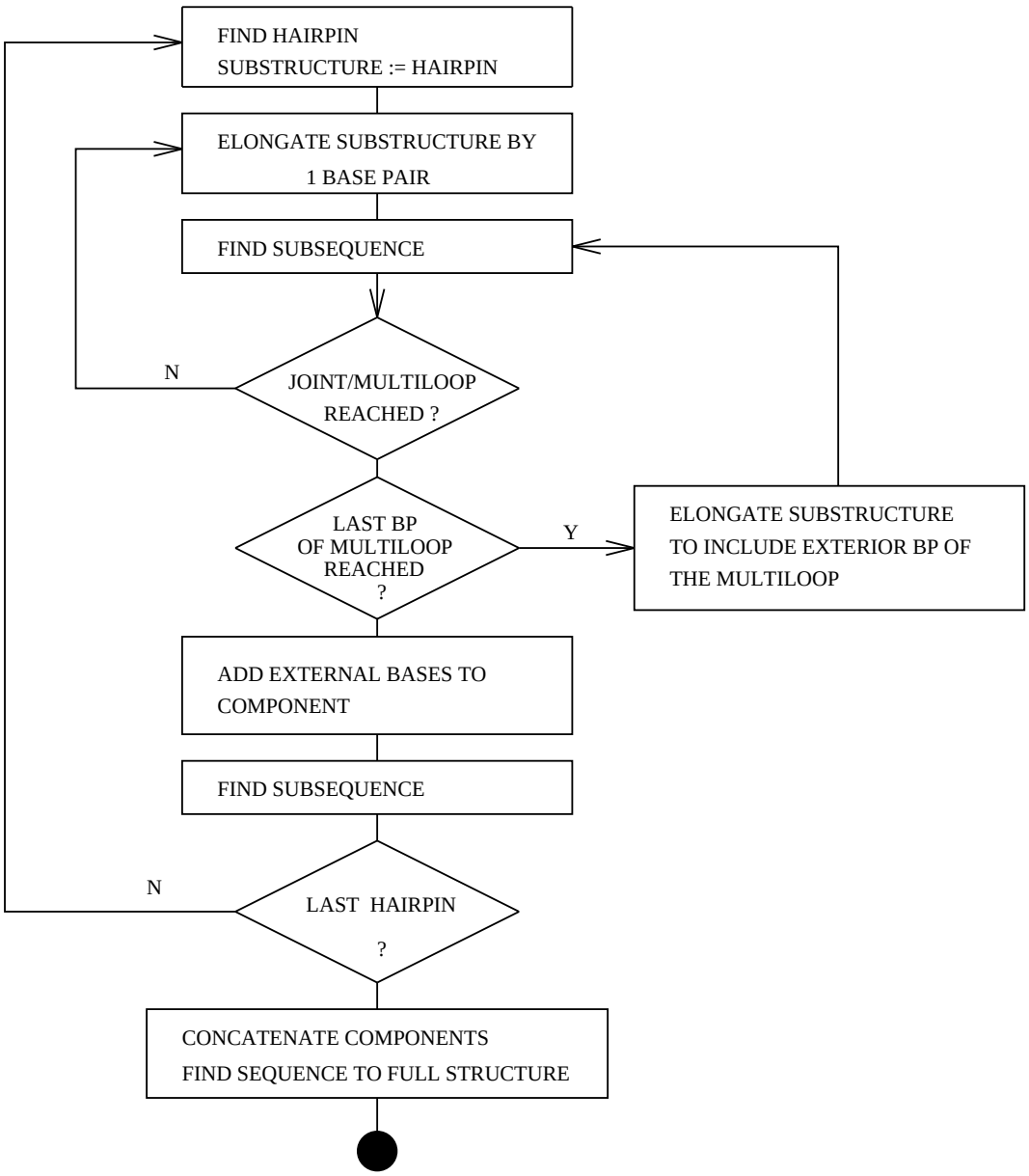


Figure 11: Flow chart of the inverse folding algorithm.

For the actual optimization (denoted by '`FindStructure`' in figure 11) we use the simplest possibility, an adaptive walk. In general, an adaptive walk will try a random mutation, and accept it if the cost function decreases. A mutation, here, consists in exchanging one base at positions that are unpaired in the target structure $\mathcal{T}$, or in exchanging two bases, while retaining compatibility, if their corresponding positions pair in $\mathcal{T}$. If no advantageous

mutation can be found, the procedure stops, and we may start again with a new initial string I_0.

To calculate the cost function of some (sub)sequence I, its minimum free energy structure $\mathcal{S}(I)$ is obtained from the fold algorithm (if appropriate under the condition that its $5'$ and $3'$ end pair) and compared to the corresponding piece of the target structure. As long as some kind of structure distance, such as the tree edit distances described previously, is chosen as cost function, the performance of the algorithm turns out to be nearly unaffected. Our implementation therefore uses the simplest possibility, a base pair distance that simply counts the number of pairs not in common between the two structures.

A different sort of cost function can be obtained form just the energies. Let $E_{min}(I)$ be the energy of the minimum free energy folding of the test sequence I and $E(I, \mathcal{T})$ the energy of the sequence I on the desired structure $\mathcal{T}$, then $E(I, \mathcal{T}) - E_{min}(I) \geq 0$ where the equality holds only if $\mathcal{T}$ is a minimum free energy structure of I. Compared to structure distances this cost function can be improved in very small increments. Using such a cost function will therefore increase the probability of finding a solution but slow down the process for relatively frequent structures since so many steps are necessary to reach that solution.

Since the calculation of $E(I, \mathcal{T})$ needs only $\mathcal{O}(n)$ steps it can easily be calculated *in addition* to a structure distance. We have therefore implemented a combination of these two cost functions, with the structure distance as primary and the "energy distance" as secondary cost function. An adaptive walk is performed with the structure distance as cost function, however, at every mutational step the energy distance is recorded as well. If no mutation with better structure distance can be found the mutation that yields the strongest decrease in energy distance, while leaving the structure distance unchanged is accepted. In effect we perform an adaptive walk relative to the primary cost function and a gradient walk relative to the second.

Even so, the number of successful adaptive walks decreases with sequence length n, restricting the usefulness of the algorithm to sequences not longer than a few hundred bases (see figure 12). This is of course to be expected since the size of the search space and the number of minimum free energy

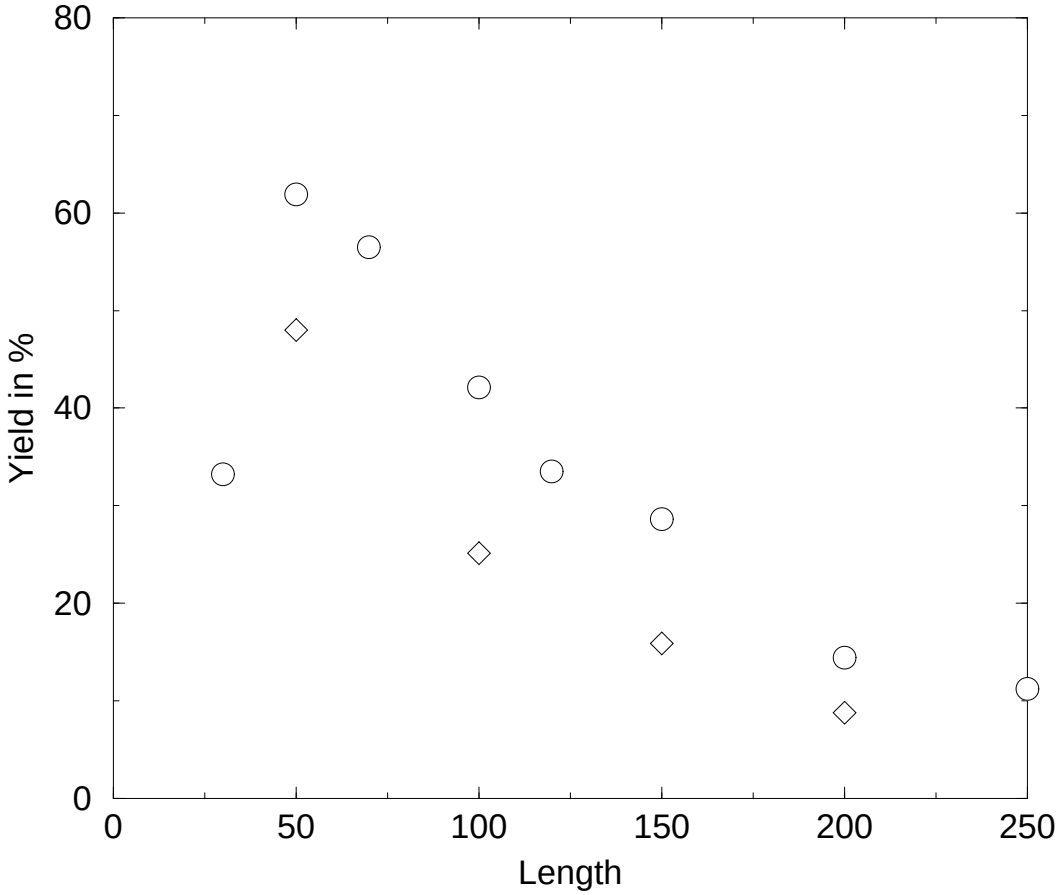


Figure 12: Mean percentage of successful inverse folds (yield) as a function of sequence length n. The algorithm is most useful for $n \lesssim 100$, otherwise it gets stuck in local optima more often. However, values for different target structures vary widely. Data are shown for **AUGC** ○ and **GC** sequences ◇.

structures both increase exponentially with n. However, for long sequences it would make more sense to search for sequences that fold into a structure sufficiently similar to the target. In other words a more coarse grained notion of structure should be used.

The efficiency of the algorithm can be further increased by dividing the positions that can be mutated into three groups. Changing positions in the test sequence I that are do not pair correctly gives the best chances of for an improvement, these mutations will therefore be tried first. The second group contains positions adjacent to these. Since only positions adjacent to

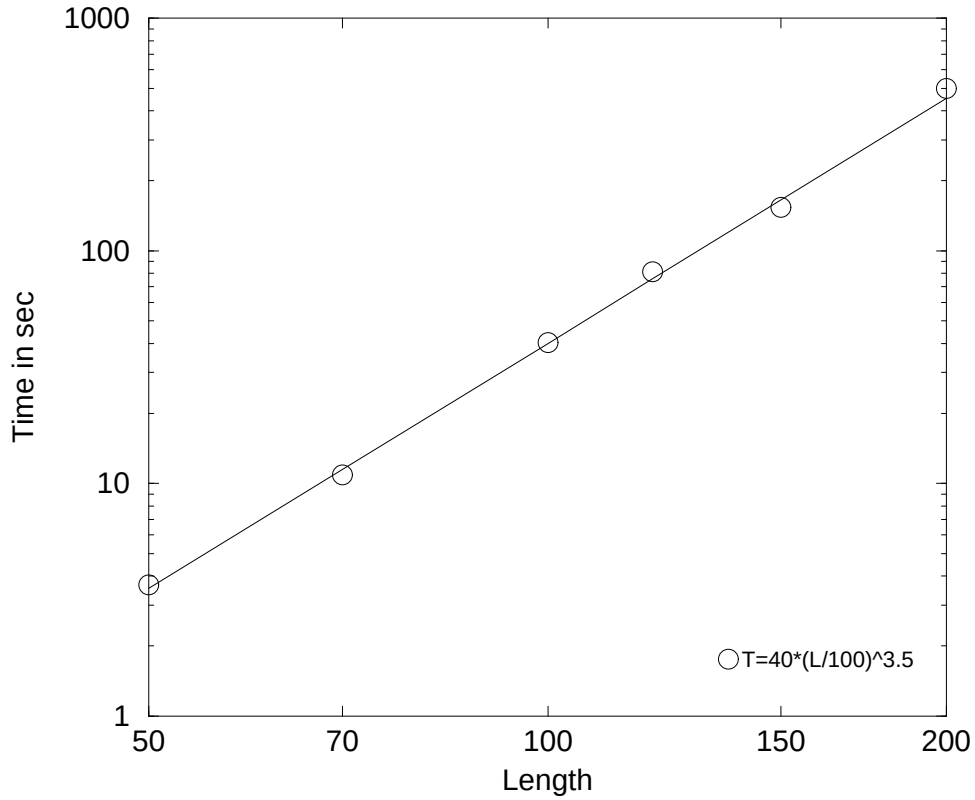


Figure 13: Performance of the inverse folding algorithms as time per sequence found. 100 structures were used at each length. Measurements were done on an Intel i860 processor. Full line: $T{=}4{\cdot}10^{-6}n^{3.5}$.

a pair give a sequence dependent contribution to the energy of the structure (through stacking energies, terminal mismatches or dangling ends), mutations in other positions will neither destabilize a wrong structural motif in the test sequences structure nor stabilize a motif of the target structure not present in the current structure (i.e. the energy distance can only increase). This third group of positions can therefore be neglected without a great risk of missing a possible solution. Other heuristics, such as a preference for GC pairs over AU pairs have been avoided, since they would bias the generated sequences.

In the present form the algorithm is usable for sequences up to a length of about a few hundred. As can be seen in figure 13 the computational cost increases more slowly than might be expected. However, the times needed per solution vary widely for different targets, so that the averages shown in the figure may be misleading. This reflects differences in the frequencies of the structures, that will be examined more closely in the next chapter. Clearly the algorithm has little chance of finding a sequence that occurs only once in the sequence space for length 50, while a frequent structure of length say 1000 may still be found in reasonable time.

Sequences generated by the inverse folding algorithm will often show many suboptimal structures with only slightly larger energies. In view of the uncertainties in the energy model this is often unsatisfactory. A variation of the inverse folding problem, therefore, searches for sequences that maximize the probability of folding into the target structure. Using the partition function algorithm $E(I, \mathcal{T}) - F(I)$ can serve as cost function, where $F(I) = -kT \ln Q(I)$ is the free energy of the Boltzmann ensemble for the sequence I. It is, however, not straight forward to do this search in a recursive manner as described above, since the optimization of substructures should only proceed up to a suitable threshold. At the moment, our implementation only does a simple adaptive walk and is thus suitable only for sequences up to about tRNA length. By using the distance of two pair probability matrices sequences with some desired equilibrium ensemble of structures could be generated as well.

3.5. Implementation of the Algorithms

3.5.1. The Vienna RNA Package

Implementations of the algorithms described above are available within the Vienna RNA Package. The package provides both stand-alone programs for folding and comparing of secondary structures as well as a library to link with other C programs. It can be obtained via anonymous ftp from `ftp.itc.univie.ac.at`.

Folding algorithms
Both folding algorithms have been integrated into a single interactive program including postscript output of the minimum energy structure and the base pairing probability matrix.

The minimum free energy algorithm uses integer arithmetic only to provide good performance even on machines without sophisticated floating point units. Assuming 32 bit integers it requires $6\,n^2$ bytes of memory to fold a sequence of length n. The three triangular matrices `F`, `C` and `FM` are stored in columns, while the calculation proceeds row-wise. Through the use of three linear auxiliary arrays it then possible to access only consecutive positions in memory in the computationally most costly parts of the program.

The five triangular matrices needed to calculate the partition function use single precision (32 bit) floating point numbers so that about $10\,n^2$ bytes are needed. In order to overcome overflows for longer sequences we re-scale the partition function of a subsequence of length ℓ by a factor $\tilde{Q}^{\ell/n}$, where $\tilde{Q}$ is an estimate of the partition function. If a minimum free energy has already been calculated $\tilde{Q} = \exp\left(1.04\,E_{min}/kT\right)$ gives a good estimate, otherwise we use:

$$\tilde{Q} = \exp\left(\frac{-184.3 + 7.27(T - 37.)}{RT}\,n\right), \tag{138}$$

where T is the temperature in °C and the Boltzmann factor RT is given in Kcal/Mol. Unless the stability of large parts of the sequence deviates very strongly from the rest the algorithm can fold sequences some thousand nucleotides long. For the minimum energy algorithm maximal length of the sequence is limited only by the available memory.

The performance of the algorithms reported here is compared with Zuker's more recent program `mfold 2.0` [78] (available via anonymous ftp from `nr-cbsa.bio.nrc.ca`) which computes suboptimal structures together with the minimum free energy structure in table 7. The computation of the minimum free energy structure and partition function including the entire matrix of base pairing probabilities is considerably faster with the present package (although we do not provide information on individual suboptimal structures). On an `IBM-RS6000/560` with 256 Mbyte of memory folding of the entire 4220 nucleotides long genome of the bacteriophage Qβ took 9.5 hours. An sample session of the `RNAfold` program is shown in figure 14.

Table 7. Performance of implementations of folding algorithms. CPU time is measured on a SUN SPARC 2 Workstation with 32M RAM. Data are for random sequences.

n	CPU time per folding $[s]$		
	RNAfold 1.0		mfold 2.0
	MFE	MFE+PF	
100	2.0	6.2	24.5
200	10.9	34.7	129.6
300	32.2	97.4	354.4
500	96.6	312.4	1258.3
690	228.1	743.9	3105.1

Because of the simplifications in the energy model and the uncertainties in the energy parameters predictions are not always as accurate as one would like. It is, therefore, desirable to include additional structural information from phylogenetic or chemical data.

```
Input string (upper or lower case); @ to quit
.........1.........2.........3.........4.........5.........6.........7........

CACUACUCCAAGGACCGUAUCUUUCUCAGUGCGACAGUAA
.((.......<<..........||............)).. 
length = 40
CACUACUCCAAGGACCGUAUCUUUCUCAGUGCGACAGUAA
.((((((..(((((.....)))))...))).....))).. 
minimum free energy = 0.83
```

a)

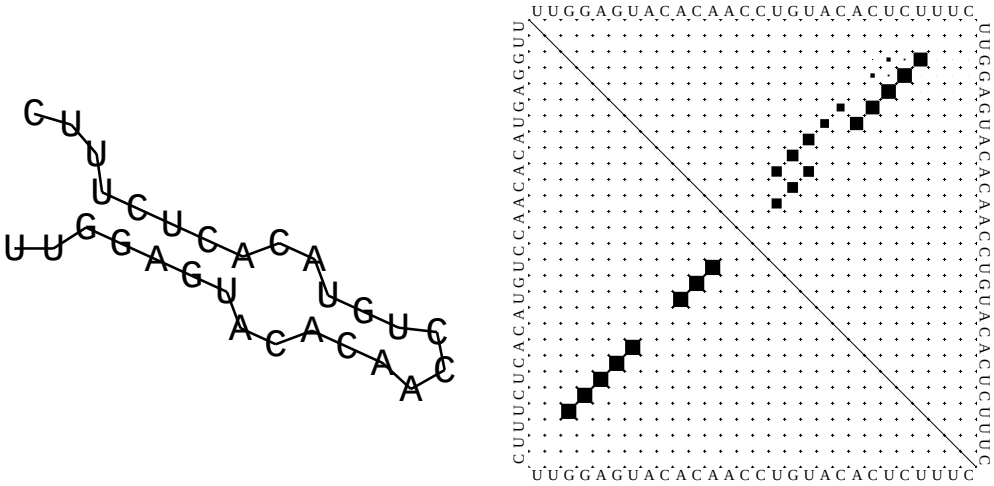
```
CACUACUCCAAGGACCGUAUCUUUCUCAGUGCGACAGUAA
((((.....(((((.....)))))...)))).........
minimum free energy = -1.52
```

b)

Figure 15: a) Example Session of `RNAfold -C`. The constraints are provided as a string consisting of dots for bases without constraint, matching pairs of round brackets for base pairs to be enforced, the symbols '<' and '>' for bases that are paired upstream and downstream, respectively, and the pipe symbol '|' denoting a paired base with unknown pairing partner. **b)** shows minimum free energy structure without constraints for comparison.

```
tram> RNAfold -T 42 -p1
Input string (upper or lower case); @ to quit
.........1.........2.........3.........4.........5.........6.........7........
UUGGAGUACACAACCUGUACACUCUUUC
length = 28

UUGGAGUACACAACCUGUACACUCUUUC
..(((((..(((...)))..)))))...
minimum free energy = -3.71
..((((((.[[,,...]]).))))))...
free energy of ensemble = -4.39
frequency of mfe structure in ensemble 0.337231
```



rna.ps dot.ps

Figure 14: Interactive example run of `RNAfold` for a random sequence. When the base pairing probability matrix is calculated the symbols `. , | { } ( )` are used for bases that are essentially unpaired, weakly paired, strongly paired without preferred direction, weakly upstream (downstream) paired, and strongly upstream (downstream) paired, respectively. Apart from the console output the two postscript files `rna.ps` and `dot.ps` are created. The lower left part of `dot.ps` shows the minimum energy structure, while the upper right shows the pair probabilities. The area of the squares is proportional to the binding probability.

Our minimum free energy algorithm allows to include a variety of constraints into the secondary structure prediction by assigning bonus energies to structures honoring the constraints. One may enforce certain base pairs or prevent bases from pairing. Additionally, our algorithm can deal with bases that have to pair with an unknown pairing partner. A sample session is described in

figure 15.

Inverse folding
The inverse folding algorithm described before is implemented in the `RNAinverse` program. Both searching for minimum free energy structures and maximization of the structures frequency in the ensemble are supported. The program can start the search with either a random sequence or one specified by the user. If the initial sequence contains lowercase characters the corresponding positions will not be modified during the optimization. Output of the program contains both the found solutions and their distance to the initial sequence.

Structure comparison
Structure distances as described in section 2.2 can be calculated using the `RNAdistance` program. The program implements both tree-edit and alignment distances with two sets of edit costs. Calculation of tree-edit distances follows the method of Shapiro and Zhang [46] but has been augmented with a backtracking that yields the optimal sequence of edit steps. From this one can construct two "aligned" trees containing "gap" nodes, analogous to conventional aligned sequences.

Distances are calculated by first converting the structures in one of four possible tree representations (full structures, HITs, loop structures and weighted loop structures), thereby offering four levels coarse graining. Output of the `RNAfold` program can be piped directly into `RNAdistance` to compare the structures of different sequences. A sample session is again shown in figure 17.

Other programs
Finally, our package contains programs to analyze distance matrices using split decomposition [79] and tree reconstruction algorithms, as well to analyze sets of sequences using statistical geometry [80]

3.5.2. Parallel Folding Algorithm

Since the computational cost of folding can be very high it is tempting to try an implementation for parallel computers. In the following we present

```
tram> RNAinverse -Fm -R -3
Input structure & start string (lower case letters for const positions)
@ to quit, and 0 for random start string
.........1.........2.........3.........4.........5.........6.........7.........
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))))....
0
length = 76
CUAUACUACGAGGAUAAUCUGCCUUUUGCCAAAGAGGGUGGCAUUUCAUCAGCUCCGAAUGCUGAGGUAUAGCGAA 20
AGCUCUGAUAUCUCUACGAAUAGAUCCUUUAUAUCUCUUAAAGCGUGUCUGGAAGAUAACUCCAGCAGAGCUUGUG 25
UUCUCCUGUAGUCGACUUUAGGACUCGAGGCCGUAUUUGCCUCACGGAAAUGUUUACAAUGCAUUAGGAGGAGUGC 29
```

A

```
tram> RNAinverse -Fp -R 3
Input structure & start string (lower case letters for const positions)
 @ to quit, and 0 for random start string
.........1.........2.........3.........4.........5.........6.........7.........
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))))....
0
length = 76
GCUAGCGUUGGGCUUUUUUUCGCCCUGCCGCAAAACCCGCGGCUUCUCGCUACAUCUCUCGUAGCCGCUAGCAAAA 50
(0.844786)
GCGUUACAAGCGCAAUCCCCCGCGCAGCGUCAAAACCCGACGCCAACAGCUACAAAACCCGUAGCGUAACGCAAAA 55
(0.859519)
GCGCGCCAAGCGCAAAAAAAAGCGCAGCCGCAAAACACGCGGCAAAAAGCGGCAGAAAAAGCCGCGGCGCGCAAAA 49
(0.85046)
```

B

Figure 16: Sample session of `RNAinverse`. A: Minimum free energy. B: Partition function.
The secondary structure of $tRNA^{phe}$ is used as target. Data on the natural $tRNA^{phe}$ sequence for comparison: the clover leaf structure occurs only with a probability of 0.17 and with even smaller probabilities for the three sequences found by `RNAinverse -Fm` (0.089, 0.033, and 0.033, respectively).

```
tram> RNAdistance -Dfh -B

Input structure; @ to quit
.........1.........2.........3.........4.........5.........6.........7 .........
((.(((((((.....)))))))..))....((..((((.....)))).)).
.....((((..((((..........)))).))))......(((....))).
f:  26
(-----(.(((--((((.....-----))))-))).))....((..((((.....)))).)).
-.....(-(((..((((..........)))).)))-)-....--.--(((....-)))-.---
h:  32
(----((U1)((U5-)P7)(U1)P2)(U4)((U2)((U5)P4)(U1)P2)(U1)R1)
((U5)((U2)((U10)P4)(U1)P4)(U5)-----((U4)P3)(U1)-------R1)
```

Figure 17: Interactive sample session of `RNAdistance`. For this example we calculated the tree-edit distance of the full and HIT representation of two random sequences folded by RNAfold.

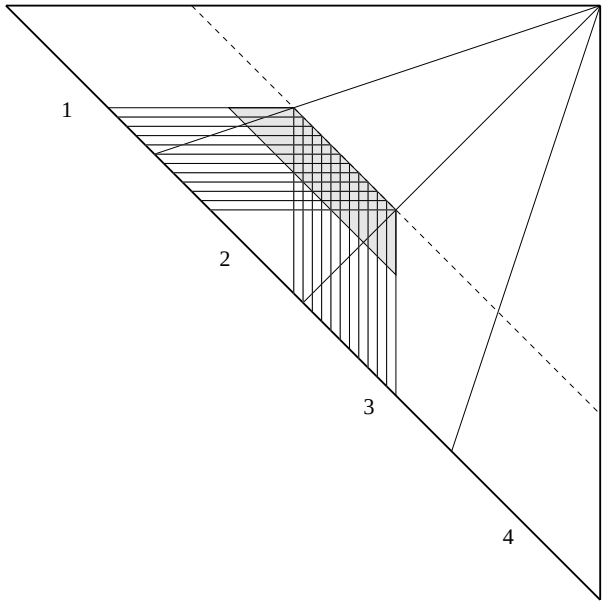


Figure 18: Representation of memory usage by the parallel folding algorithm. The triangle representing the triangular matrices `F`, `C`, and `FM`, respectively, is divided into sectors with an equal number of diagonal elements, one for each processor. The computation proceeds from the main diagonal towards the upper right corner. The information needed by processor 2 in order to calculate the elements of the dashed diagonal are highlighted. To compute its part of the dashed diagonal processor 2 needs the horizontally and vertically striped parts of the arrays `F` and `FM`, and the shaded part of the array `C`. The shaded part does not extend to diagonal, because we have restricted the maximal size of interior loops.

a parallelized version of the minimum energy folding algorithm for message

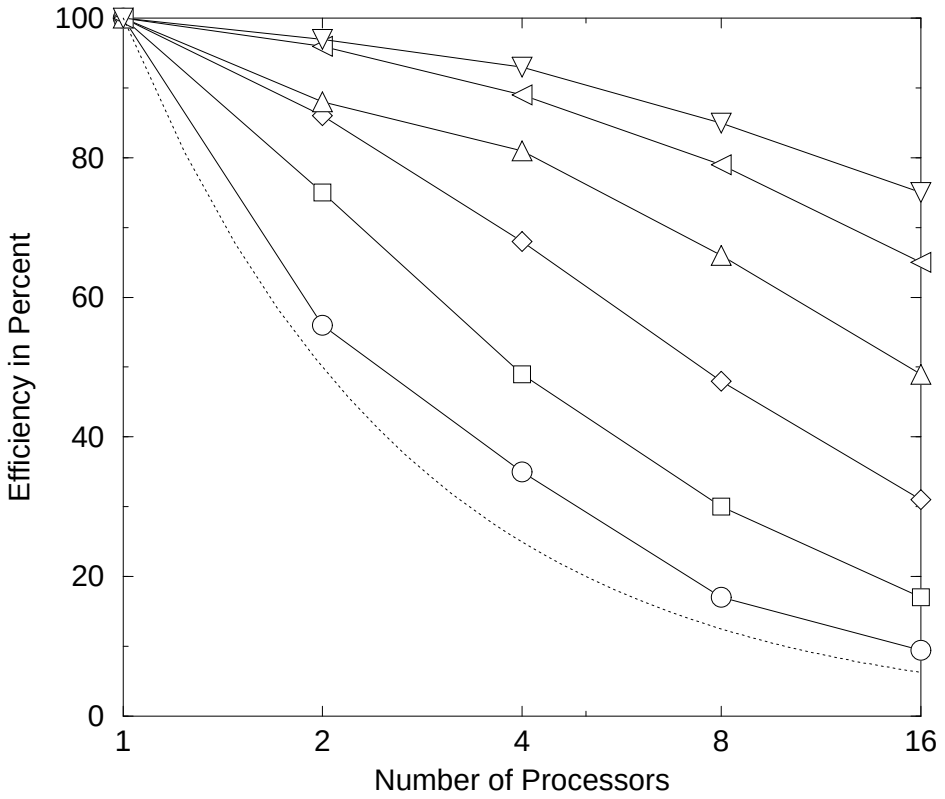


Figure 19: Performance of parallel algorithm for random sequences of length 50 ○, 100 □, 200 ◇, 400 △, 700 ◁, 1000 ▽. Efficiency is defined as speedup divided by the number of processors. The dotted line is $1/n$ corresponding to no speedup at all.

passing systems. Since all subsequences of equal length can be computed concurrently, it is advisable to compute the arrays `F`, `C` and `FM` in diagonal order, dividing each sub-diagonal into P pieces. Figure 18 shows an example for 4 processors. Our algorithm stores the arrays `F` and `FM` twice once in columns and once in rows, while the `C` array is stored in diagonal order. However, each processor holds only those values needed to evaluate its section of the current sub-diagonal. Since the number of rows and columns that have to be stored decreases while their length increases, the maximal memory requirement occurs at $d = n/2$, where we need $n^2/(2P)$ integers each for `F`

and `FM`. Because we restrict the maximal loop size, the array `C` needs only $\mathcal{O}(n)$ storage. After completing one sub-diagonal each processor has to either send a row to or receive a column from its right neighbor, and it has to either receive a row from or send a column to its left neighbor.

Since we do not store the entire matrices, we cannot do the usual backtracking to retrieve the structure corresponding to the minimum energy. Instead, we write for each index pair (i, j) two integers to a file, which identify the term that actually produced the minimum. The backtracking can then be done with $\mathcal{O}(n)$ readouts. All in all the algorithm therefore needs $\mathcal{O}(n)$ communication and I/O steps, with each communication step transferring $\mathcal{O}(n)$ integers, while the computational effort is $\mathcal{O}(n^3)$. The communication overhead therefore becomes negligible for sufficiently long sequences.

On an Intel `Ipsc/860` the I/O overhead becomes negligible for sequences longer than some 200 nucleotides. The efficiency of the parallelization as a function of sequence length and number of processors can be seen in figure 19.

The partition function algorithm can in principle be parallelized analogously.

4. The RNA Folding Map

4.1. Statistics of Structural Elements

In this section we present statistics of RNA secondary structures generated by folding random sequences. Such statistics can serve as a reference to compare structures of specific sequences with a random sample as well as to compare different folding algorithms. To generate such a statistical reference we folded samples of typically 10^5 random sequences from the **AUGC**, **AU**, **GC** alphabets for lengths up to 1000. Results for the different alphabets were then compared with each other and with random structures as defined in chapter 2. The random structures used here are constrained to have a minimum hairpin size of 3 and a minimum stack size of 2, i.e., there are no isolated pairs. The structures are compatible with sequences with a stickiness $p = 1, \frac{3}{8}, \frac{1}{2}$ corresponding to alphabets in which all bases can pair, the biophysical **AUGC** and a binary (e.g. **AU** of **GC**) alphabet.

• *Number of Base Pairs*

The mean number of base pairs increases linearly with sequence length n in all cases. As expected minimum free energy structures show somewhat more pairs than random structures with the same stickiness, **AU** sequences produce more pairs than **AUGC** because of the higher stickiness in spite of the lower stability of **AU** pairs. Structures on the **GC** alphabet not only show much more pairs but a remarkably narrow distribution.

• *Number of Loops and Stacks*

The number of loops must equal the number of stacks because every loop must be closed by a stack. Even for short chain lengths the mean number of loops and stacks scales linearly with n. Because loops are destabilizing folded structures form significantly fewer loops than random structures. Dependence on the alphabet is weak. In particular **AUGC** and **GC** sequences form nearly the same number of loops and stacks.

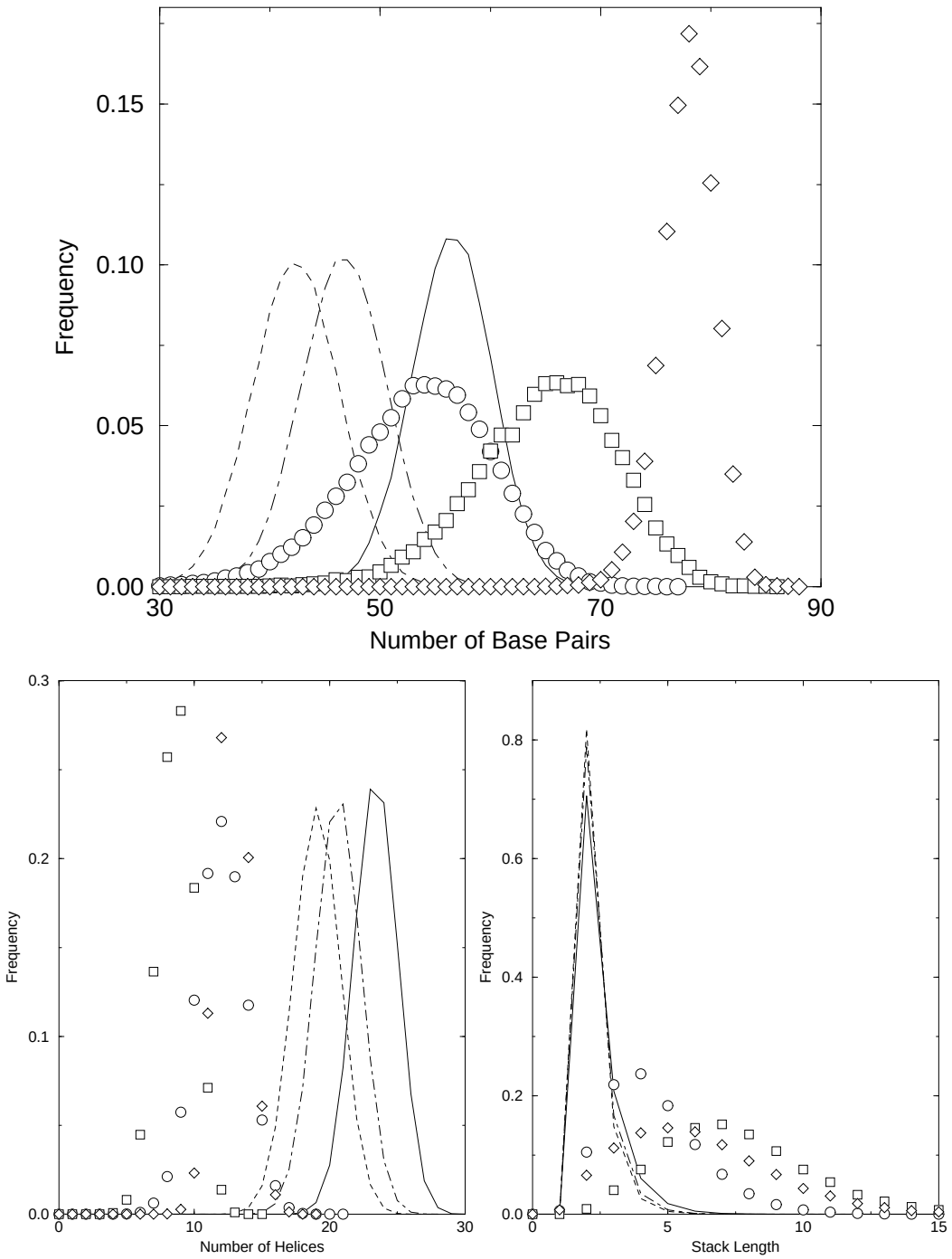


Figure 20: Distribution of the number of base pairs (above), the number of stacks and loops (lower left) and the Distribution of stack sizes (lower right) for chain length n=200. Full lines denote random structures with stickiness p=1, dashed lines stickiness $p=\frac{1}{2}$ corresponding to binary alphabets dashed dotted lines stand for stickiness $p=\frac{3}{8}$ as in the **AUGC** alphabet. Data for folded random sequences are shown with circles ○ for the **AUGC** alphabet, squares □ for **AU** and diamonds ◇ for **GC** sequences.

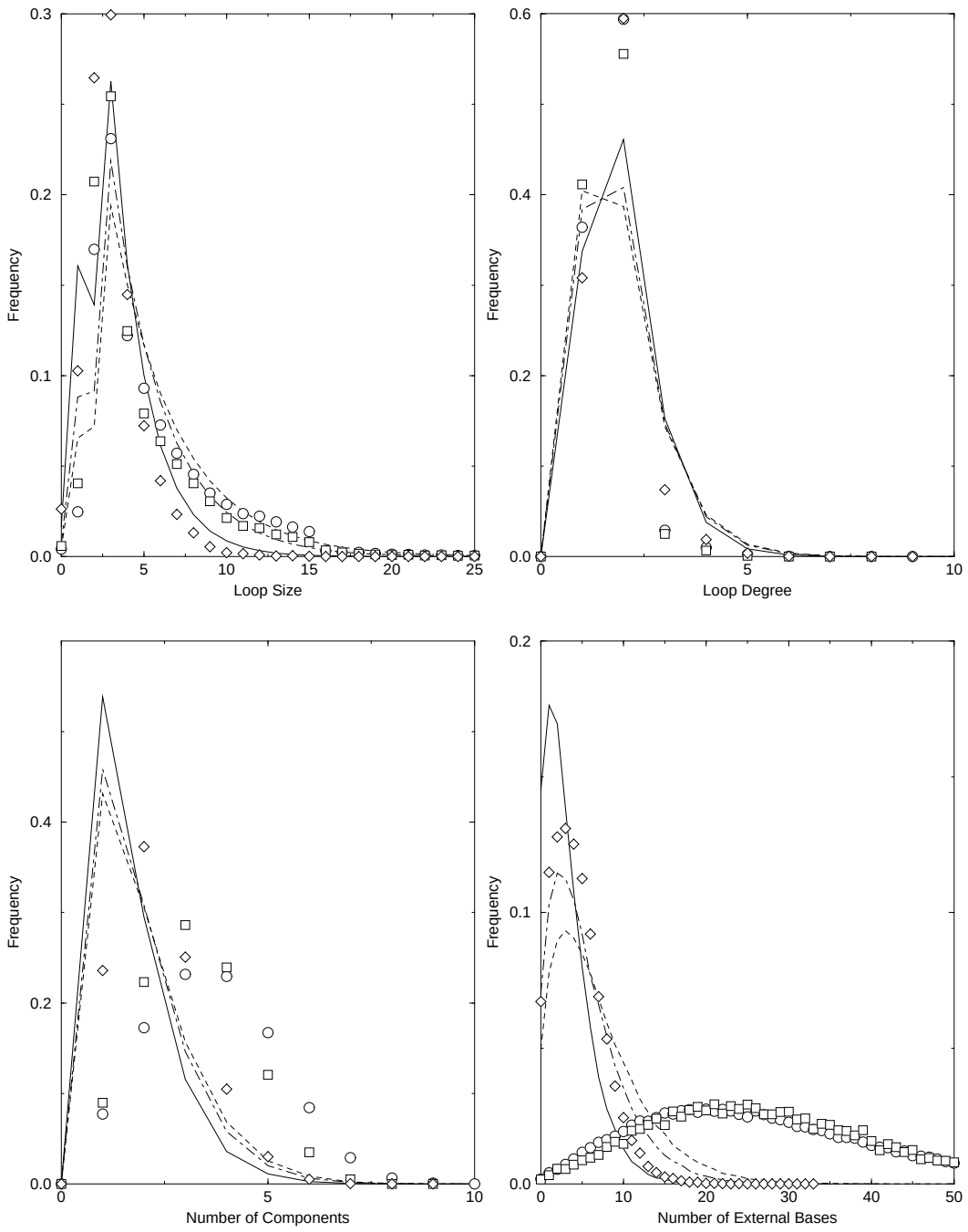


Figure 21: Distribution of loop sizes (upper left), loop degrees (upper right), components (lower left) and external digits (lower right) for random structures with stickiness p=1, $\frac{1}{2}$, $\frac{3}{8}$ and folded random sequences form the **AUGC**, **AU**, and **GC** alphabets.

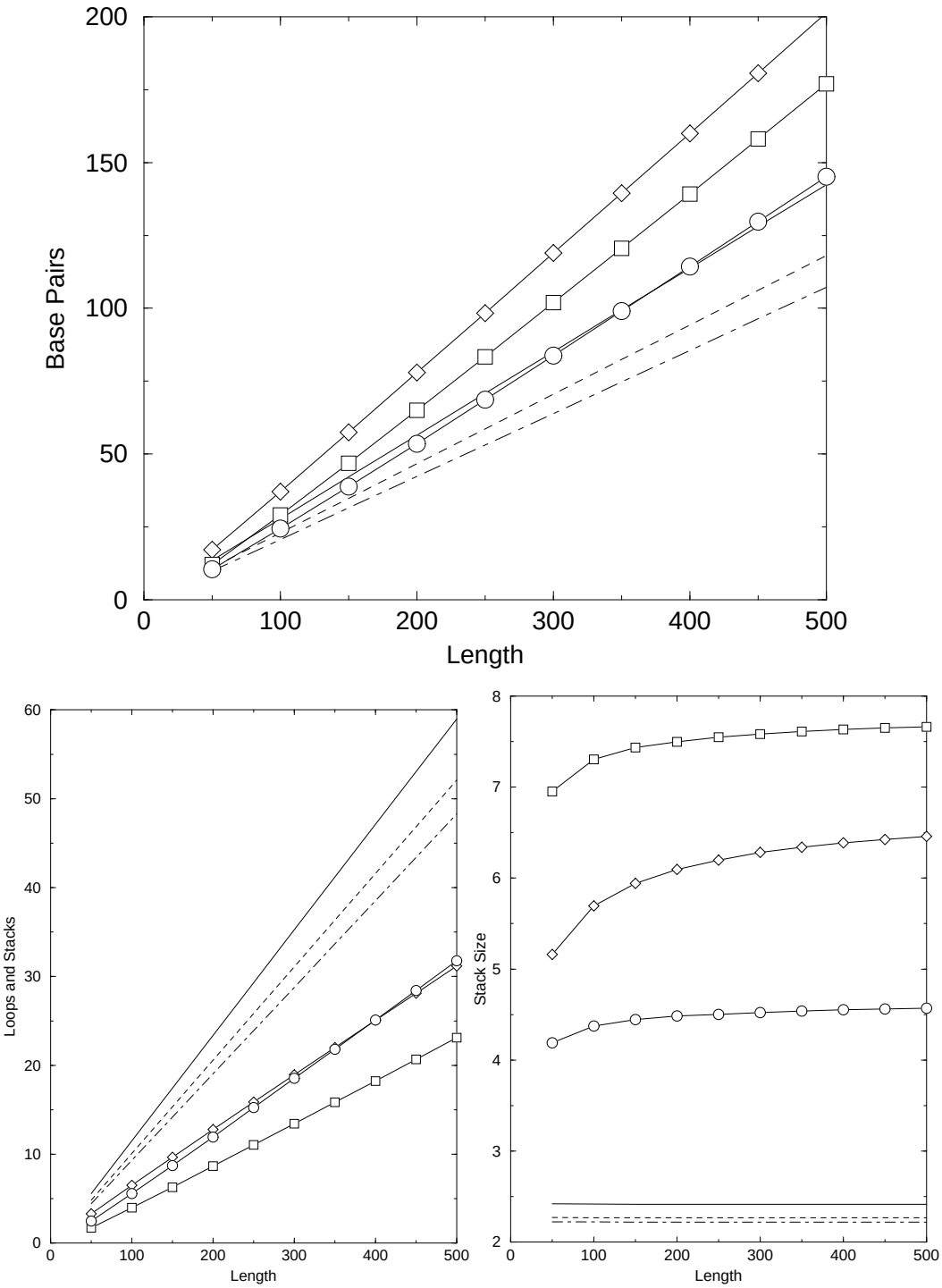


Figure 22: Mean values of the number of base pairs (above), the number of stacks and loops (lower left) and of stack sizes (lower right) versus chain length. Data for folded random sequences are shown as lines with symbols. Full lines, dashed lines and dash dotted lines denote random structures with stickiness p=1, $\frac{1}{2}$, $\frac{3}{8}$. As before circles ○ stand for the **AUGC** alphabet, squares □ for **AU** and diamonds ◇ for **GC** sequences.

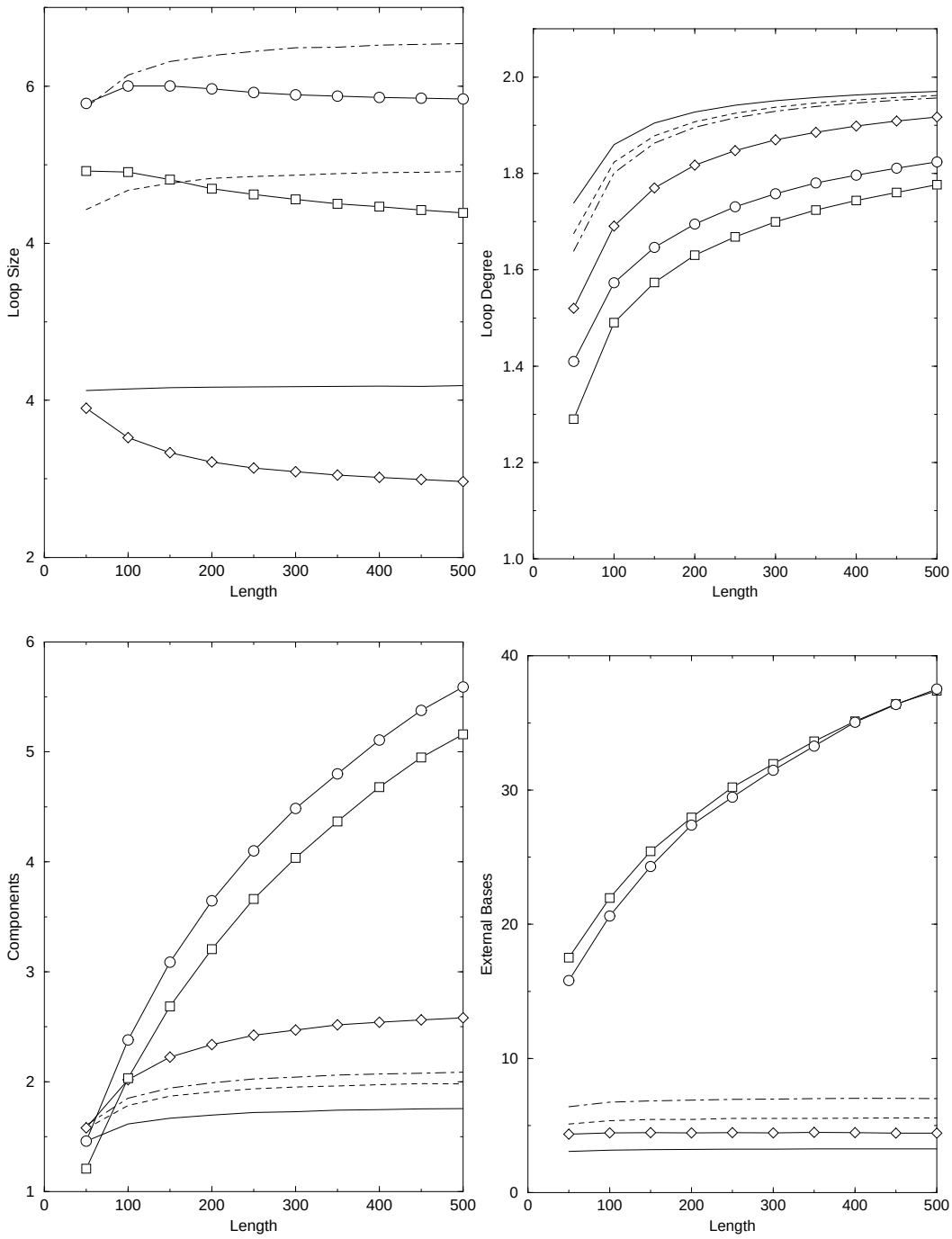


Figure 23: Mean values for loop sizes (upper left), loop degrees (upper right), components (lower left) and external digits (lower right) as a function of chain length. Data for random and folded structures are marked as previously.

• *Stack Size*

The mean stack size of random structures converges to a constant value at very small chain lengths. For minimum free energy structures, especially from the **GC** alphabet convergence is much slower. The most important stabilizing contributions to a structure come from the stacking of base pairs. Minimum free energy structures therefore exhibit longer stacks than random structures. While the distribution is very similar for random structures of different stickiness, folded structures need more pairs to stabilize a stack for alphabets with weak pairs such as **AU**.

• *Loop Size*

The average mean loop size converges at moderate chain lengths to a constant value. The distributions are similar for all alphabets as well as for random structures. This may be because the destabilizing energy of loops increases only logarithmically with size, so that loop sizes have a much smaller effect on the energy than stack sizes. The slight step in the distributions at loop size 15 is an artifact because a maximum size for interior loops of 15 was used in the fold algorithm.

• *Loop Degree*

Since for every additional multi loop with degree n there must be n-1 additional hairpins (with degree 1), the mean loop degree of a structure with N_l loops and N_c components is $2 - N_l/N_c$. We expect that $N_l/N_c \to 0$ for $n \to \infty$ and therefore, the average loop degree to converge to 2 in all cases. However convergence seems to be very slow. The distributions are similar for folded structures, **GC** sequences produce slightly more multi loops than other alphabets. All minimum energy structures show significantly fewer multi loops than the random structures. Prediction of multi loops is, however, the weakest point in the folding algorithm.

• *Number Components*

Combinatorics for the number of components resemble that of loop degree. Therefore they behave similar for random structures. In folded structures additional components do not incur an energy cost. Their number of components is, therefore, somewhat larger than the mean loop degree and larger

than in random structures. The difference between folded and random structures is again smallest for **GC** sequences. The mean converges to a constant value for random structures and maybe for **GC** sequences. The asymptotic behavior for **AUGC** and **AU** sequences is not clear.

• *Number of External Digits*

External digits resemble loop sizes for random sequences. In contrast to unpaired bases in a loop external digits are not destabilizing. Folded structures, therefore, show much more external digits. Remarkably, the distributions for the **AUGC** and **AU** alphabet are nearly identical, while structures from **GC** sequences are closer to random structures. The mean values converge very fast to constant values for random and **GC** structures.

4.2. Frequencies of Structures

As shown in chapter 2 the number of possible secondary structures is much smaller than the number of sequences for any sequence length n. As a crude estimate for the number of minimum free energy structures one could use the number of possible structures isolated base pairs and with at least three unpaired bases in hairpin loops. As shown before we find approximately

$$\Psi_n^{(2,3)} \approx 1.4848 \times n^{-3/2} \, (1.8488)^n \tag{139}$$

while the number of sequences in an alphabet of size κ is κ^n. Presumably, this estimate is still much to high since many of these structures are unstable on any sequence. From the statistics of base pairs shown in the previous section one can also derive a lower limit to the number of free energy structures.

Lemma. Given an alphabet $\mathcal{A}$ of size κ with stickiness p, let $\#\Sigma$ be the number of sequences of length n compatible with some structure $\mathcal{S}$ with b base pairs. Then

$$\#\Sigma = \kappa^n \cdot p^b. \tag{140}$$

Proof. The sequence at the unpaired positions is arbitrary, however, for each pair in $\mathcal{S}$ only a fraction p of possible sequences is compatible. ∎

From the data presented in the previous section we conclude that for large chain length n there are on average $\gamma[\mathcal{A}]n$ base pairs in minimum free energy structures.

Theorem. Suppose there are constants c and γ with $0 < c, \gamma < 1$ such that at least a fraction c of all sequences forms a minimum free energy structure with at least γn base pairs. Then at least

$$c \cdot \left(p^{-\gamma}\right)^n \tag{141}$$

different structures are minimum free energy structures for some sequence in sequence space.

Proof. There must be at least $c\kappa^n/\#\Sigma(\gamma n)$ different structures since from the $c\kappa^n$ sequences forming γn or more base pairs at most $\#\Sigma(\gamma n)$ sequences can be compatible with a single structure. Hence, there are at least $c\kappa^n \cdot \kappa^{-n} \cdot p^{-\gamma n}$ different structures. ∎

Our numerical results confirm that such c and γ exist. Since the distribution of the number of base pairs (see fig. 21) is approximately symmetric we find that for $c = 1/2$ $\gamma = \gamma[\mathcal{A}]$ is the expected number of base pairs n_{BP}.

Table 8. Parameters for the lower bound on the number of minimum free energy structures

Alphabet	$\gamma = n_{\mathrm{BP}}/n$	$p^{-\gamma}$
GC	0.403	1.32
AU	0.354	1.28
GCAU	0.290	1.33

The above estimate uses the maximum number of sequences that may fold into a single structure, which can only be appropriate for the most stable structures. On the other hand, equation (139) counts the number of syntactically admissible structures irrespective of their stability. Since many sequences must fold into identical structures, the question arises how these relatively few structures are distributed over sequences. The large difference between the upper and lower bound already suggests that relative frequencies of different structures may vary widely.

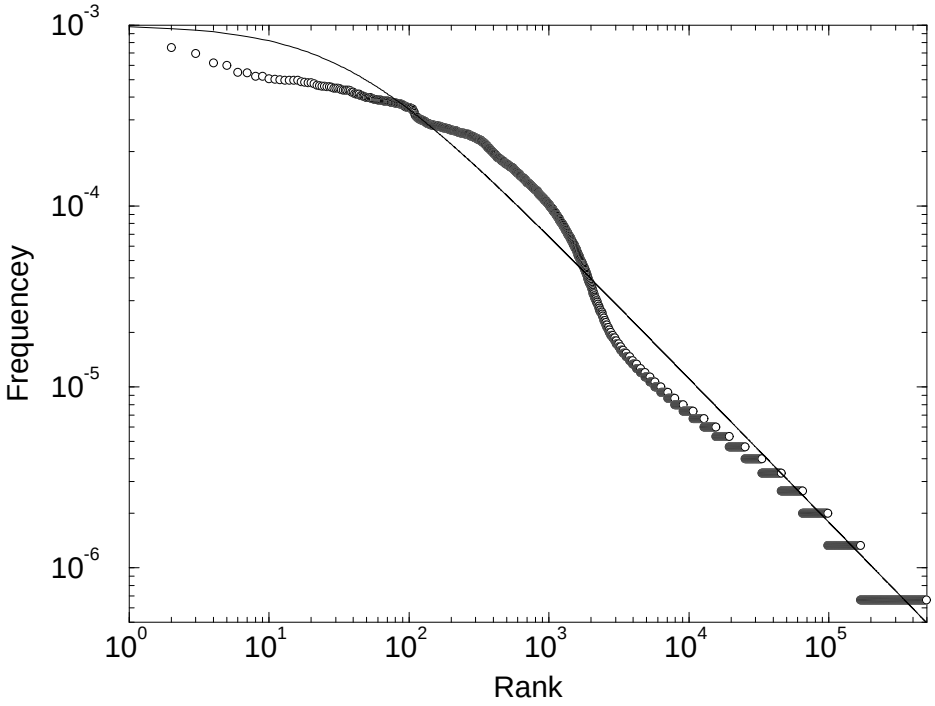


Figure 24: Frequency distribution of sequences folding into identical secondary structures for the **AUGC** alphabet with chain length n=40 . The open structure occurs is not included in the data, it occurs with a frequency of about 3%. The full line is the function $f(r)$=0.001/(1+r/32)$^{8.1}$.

To determine the frequency distribution of secondary structures one can folded large pools of random sequences, sort the resulting structures by frequency and plot the rank of each structure versus its frequency. Using the secondary structures at full resolution this can be done only for very short

($\lesssim 40$) sequences since for longer sequences one will not find any identical structures in the sample.

For longer sequences we can only study the frequencies of more coarse grained structures. Up to lengths of $n \approx 100$ using the loop structures works well for even longer sequences branching structures can be used.

The frequency distributions gathered for different lengths of sequences and levels of coarse graining are remarkably similar, following roughly the generalized form of Zipf's law given by Mandelbrot [81, 82],

$$f(r) = a(1 + r/b)^{-c}, \tag{142}$$

where r is the rank (by frequency) of the structure S and $f(r)$ is the fraction of occurrences of S in the sample. Zipf's law was originally derived from the analysis of the frequency of words in literary texts [83] and has since been found in a variety of contexts (e.g. [84]). The form given above can be derived analytically for simple models of random text [85, 86]. Zipf's law suggests that most sequences fold into few very common structures while most structures are extremely rare. Because of the asymptotic power law a constant fraction (about $0.5^{1/c}$) of structures will occur only once in a sample regardless of sample size. In of its simplest definitions Zipf's law states that about half the objects being counted will occur only once. This corresponds to $c = 1$ in the above form.

In the above parameterization of Zipf's Law the exponent c describes the distribution of rare sequences, the constant b is a rough measure for the number of frequent structures, while a gives the frequency of the most common structures. Our data suggest that there are few frequent structures and many very rare ones. The few thousand most common structures cover already more than 90% of the sequence space for **AUGC** sequences with chain length $n = 30$ and 40.

The parameters b and c depend strongly on the chain length. Not unexpectedly, the number of the most frequent structures, b, increases exponentially with chain length, see figure 27a. The parameter c describing the scaling of the power law tail of the distribution decreases with chain length, indicating that a larger fraction of sequences folds into rare structures for longer chains. The data in figure 27b are consistent with $c \to 1$ for $n \to \infty$.

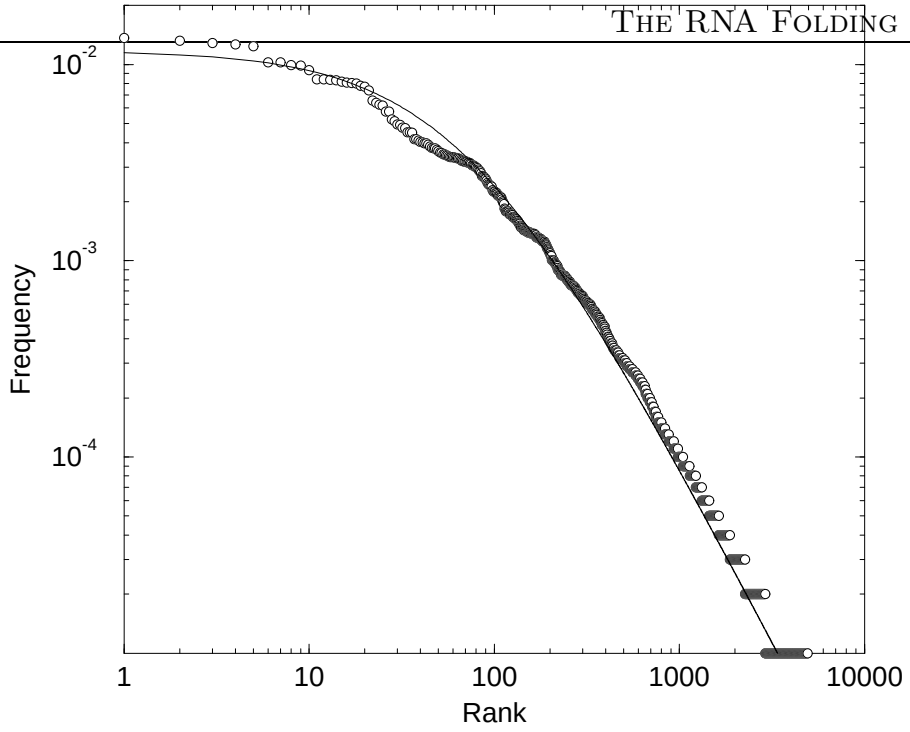


Figure 25: Distribution of loop structure frequencies for the **AUGC** alphabet with chain length n=100. The full line is a fit to equation (141).

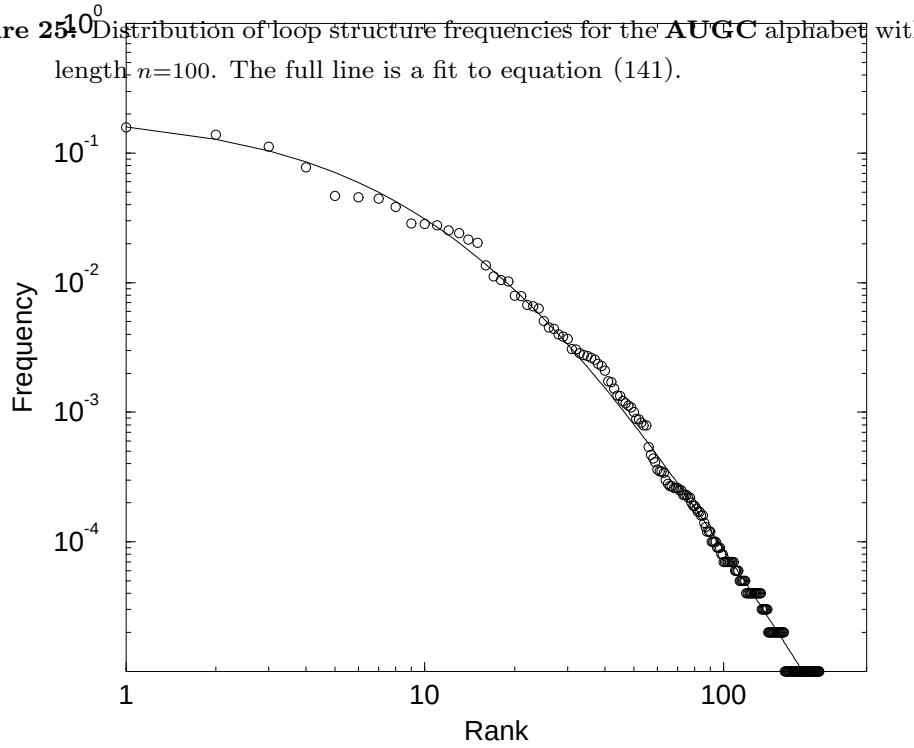


Figure 26: Distribution of branching structure frequencies for the **AUGC** alphabet with chain length n=200. The full line is a fit to equation (141) $f(r)=0.2/(1+x/16.8)^4$.

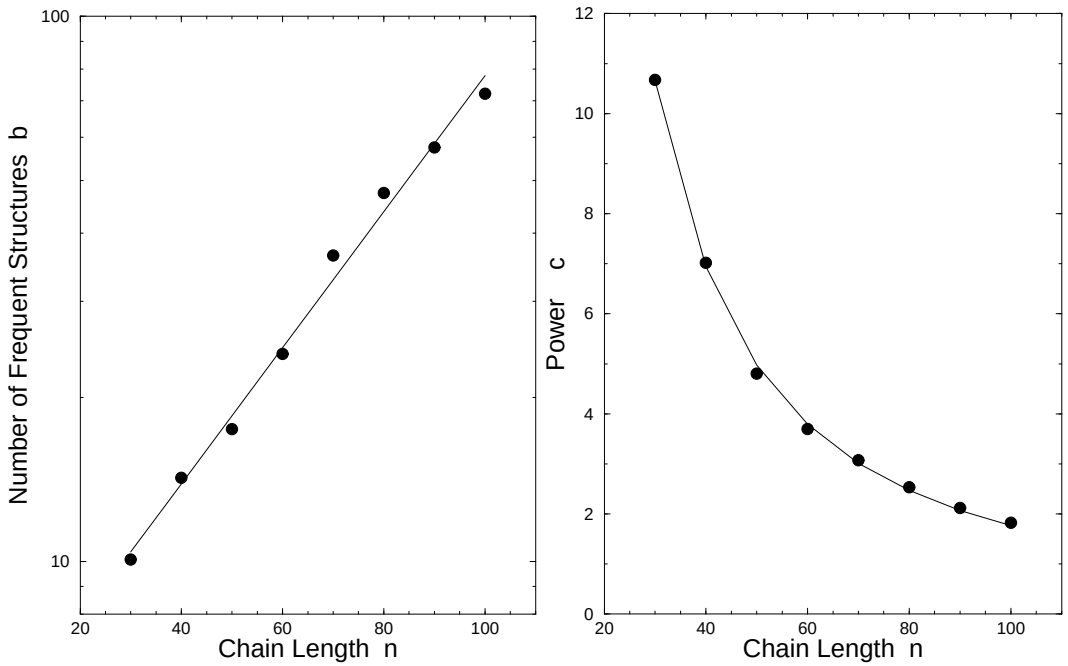


Figure 27: Dependence of the parameters of the generalized Zipf's law, equation (142), on the chain length. Data are for loop structures from the **AUGC** alphabet.

4.3. Distribution of Structures in Sequence Space

A question related to the relative frequencies of structures is their special distribution over sequence space. To study this question one needs a sample of sequences folding into the same structure that can be generated using the inverse fold algorithm.

Analyzing the pair distances of the resulting sequences shows that such sequences are distributed all over sequence space. Given that all structures that fold into a given structure have to be at least compatible with that structure, the resulting distribution can not be binomial as for true random sequences but is nearly indistinguishable from the distribution expected for random sequences compatible to the given structure. In the example shown in figure 28 random structures compatible to $tRNA^{phe}$ structure without any **G-U** pairs were used as starting point for the inverse algorithm. The resulting distribution of the solution sequences is nearly identical to that of the start sequences.

A subset of 100 sequences with $tRNA^{phe}$ structure was also compared to a

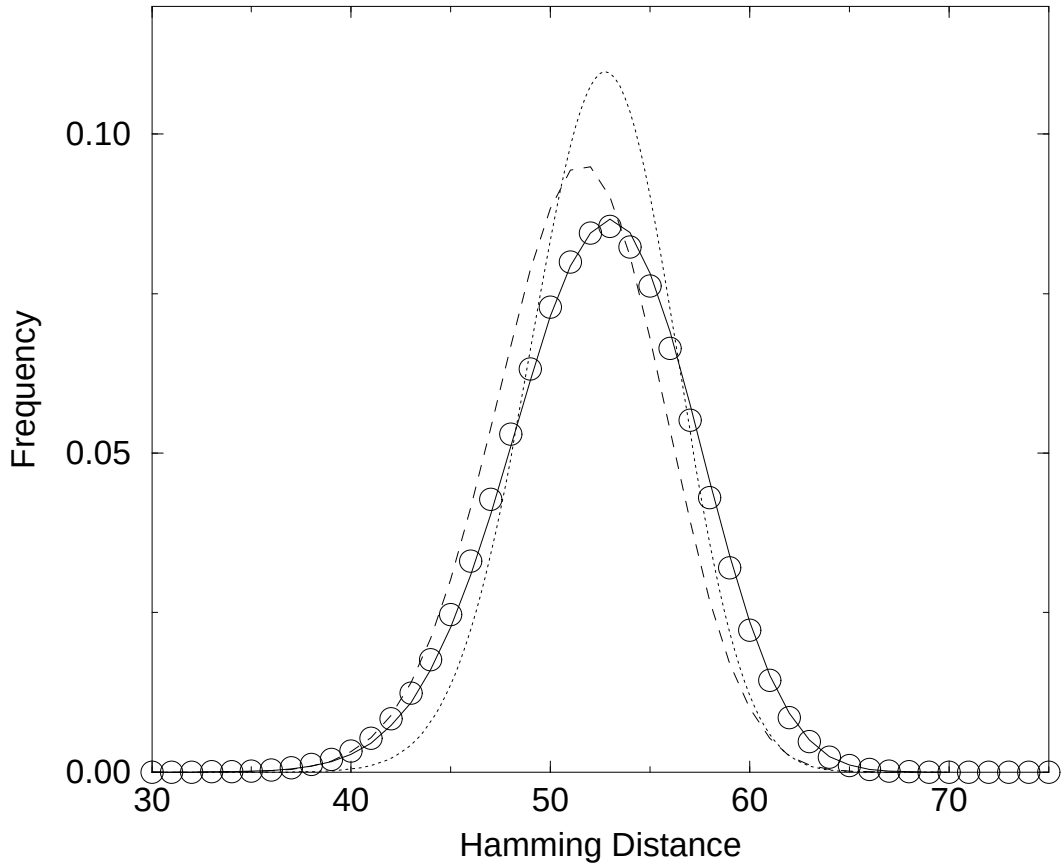


Figure 28: Distribution of 2621 tRNA analoga obtained from reverse folding ○. The start sequences did not contain **GU** pairs, and only very few **GU** pairs are found in the rRNA analoga. For comparison the distance distribution functions for random sequences (dotted line), random compatible sequences (dashed line), and random compatible sequences without **GU** pairs are shown. This indicates that sequences are randomly distributed (subject to the constraint of being compatible with the structure).

random compatible sequences using statistical geometry [80, 87] and split decomposition [79]. Both methods will detect clustered or hierarchically related data. While statistical geometry compares sequences position-wise, split decomposition analyses the matrix of pair distances. In both cases the sample of inverse folded sequences was indistinguishable from the random sample.

4.4. Correlation and Density Surfaces

A basic property of combinatory landscapes is *ruggedness*. A landscape is rugged if it has lots of local optima, if adaptive (up-hill) walks are short, and if the correlation between nearest neighbors is small. Adaptation and optimization is harder on more rugged landscapes. While the notions of local optima and adaptive walks do not have counter-parts in general combinatory maps (their definition require the image set to be ordered), the definition of pair-correlation can be generalized to mapping from one metric space into another one [34]:

$$\rho(d) = 1 - \frac{\langle D^2(f(x), f(y))\rangle_{d(x,y)=d}}{\langle D^2(f(x), f(y))\rangle_{\text{random}}} \tag{143}$$

The average in the enumerator runs over all pairs of configurations with fixed Hamming distance d while the average in the denominator runs over all pairs of configurations. Correlation functions often fall of exponentially, it is therefore convenient to define an empirical correlation length ℓ from $\rho(\ell) = 1/\mathrm{e}$. This correlation length, possibly scaled by the diameter of the configuration space, forms a measure of ruggedness, which can be used to compare landscapes. A more detailed representation of a combinatory map is given by the *density surface*. The structure density surface $P(D|d)$ is the conditional probability that two secondary structures have distance D in shape space provided their underlying sequences have distance d in sequence space. Density surfaces contain more information then the autocorrelation functions. For instance, the probability for finding a neutral neighbor, i.e., a sequence in Hamming distance 1 which folds into the same structure is $P(0, 1)$. In fact, the autocorrelation function can be obtained from the density surface:

$$\rho(d) = 1 - \frac{\sum_D D^2 P(D|d)}{\sum_d \sum_D D^2 p(d) P(D|d)} \tag{144}$$

where $p(d)$ is the probability for two randomly chosen sequences to have distance d.

The structure density surface for **AUGC** sequences of length $n = 100$ using tree edit distances of full structures is shown in figure 29. The corresponding

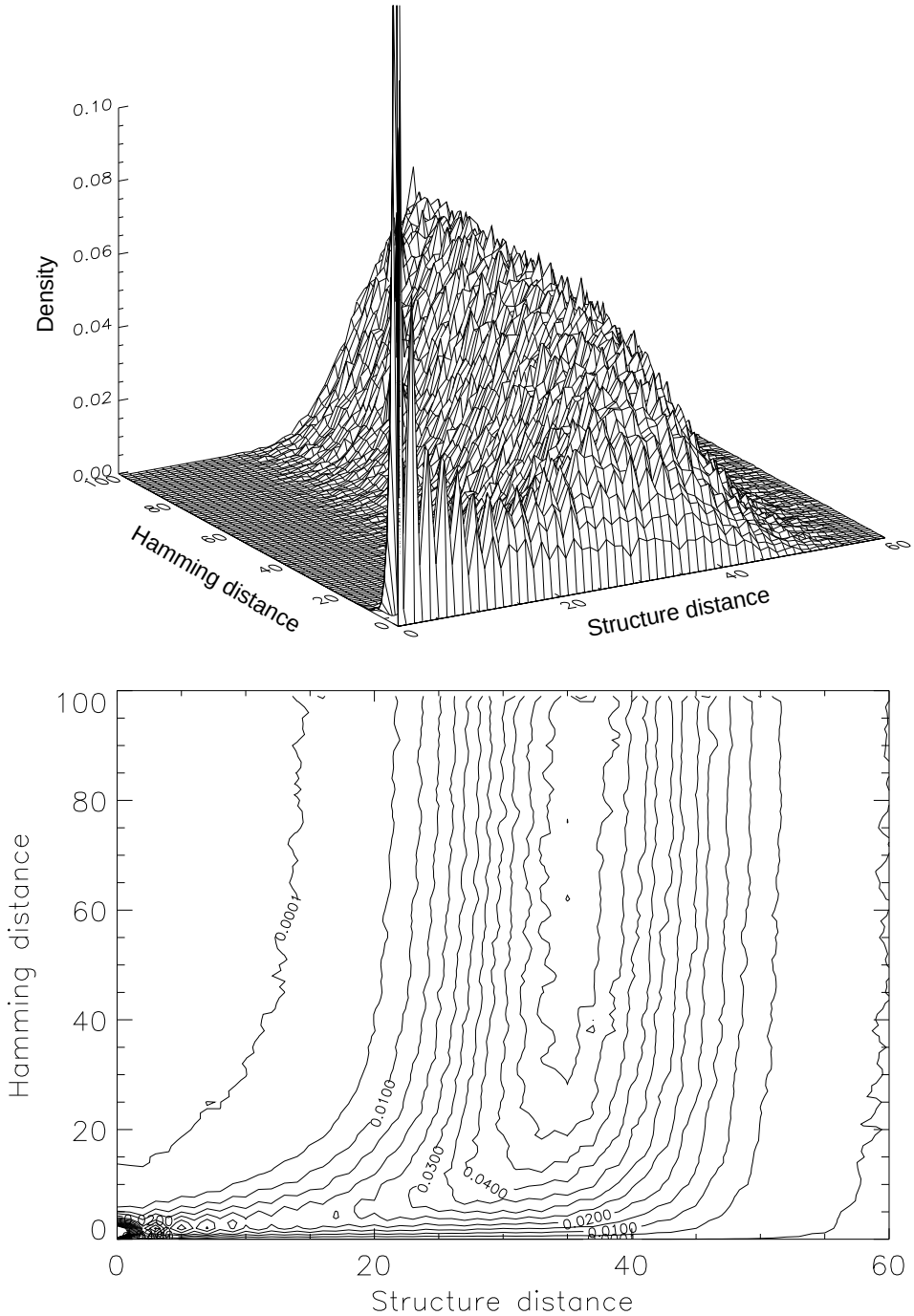


Figure 29: The structure density surface $P(D|d)$ of natural **AUGC**-sequences of chain length n=100. The density surface (upper part) is shown together with a contour plot (lower part). In order to dispense from confusing details the contour lines were smoothened. In this computation a sample of 1000 reference sequences was used which amounts to a total sample size of 10^6 individual RNA foldings.

correlation length $\ell = 7.6$ is small compared to the sequnece length. Similar calculations were performed in Pauline Hogeweg's group [88] using alignment distances. Yet, the qualitative features of the density surface were unchanged. The shape of the SDS also changes very little with sequence length, except for rescaling of the axis. Some basic properties of the folding map become apparent in the density surface plot: For very small Hamming distances ($h = 1, 2, 3$) the most probable structures are identical or very similar to the reference structure, there is nonetheless some probability that even a single mutation substantially alters the structure; beyond a distance of $h \approx 3$ identical or even closely related structures are extremely unlikely; at a distance of about 15% to 20% of the chain length the density becomes independent of h, thus approaching essentially what is expected for a sample of randomly drawn sequences ($h \approx 0.75n$). The latter indicates that the structures of a reference sequence and of its mutants at distances of 20% or larger are effectively uncorrelated, i.e memory of the reference structure is sufficiently lost to allow the mutants at that distance to acquire the essential features of any frequent minimum energy structure. A fairly small ball in sequence space already shows the global characteristics of the entire folding map f.

4.5. Shape Space Covering

For any evolutionary optimization it is of prime importance how big a volume in sequence space has to be searched in order to find a sequence with the desired properties. We may therefore pose the question how close to some given starting sequence a preselected secondary structure can be found. Stated differently the question is what radius a ball in sequence space must have to contain most common structures. This radius is called the shape space covering radius h_c. A "common" structure here is one that is not ranked in the power law tail of Zipf's distribution. This is easily ensured by using structures produced by folding a random sequence in the computer experiments below .

The fact that the SDS becomes independent of h at relatively small h is a first indication that this distance might be much smaller than average distances in sequence space.

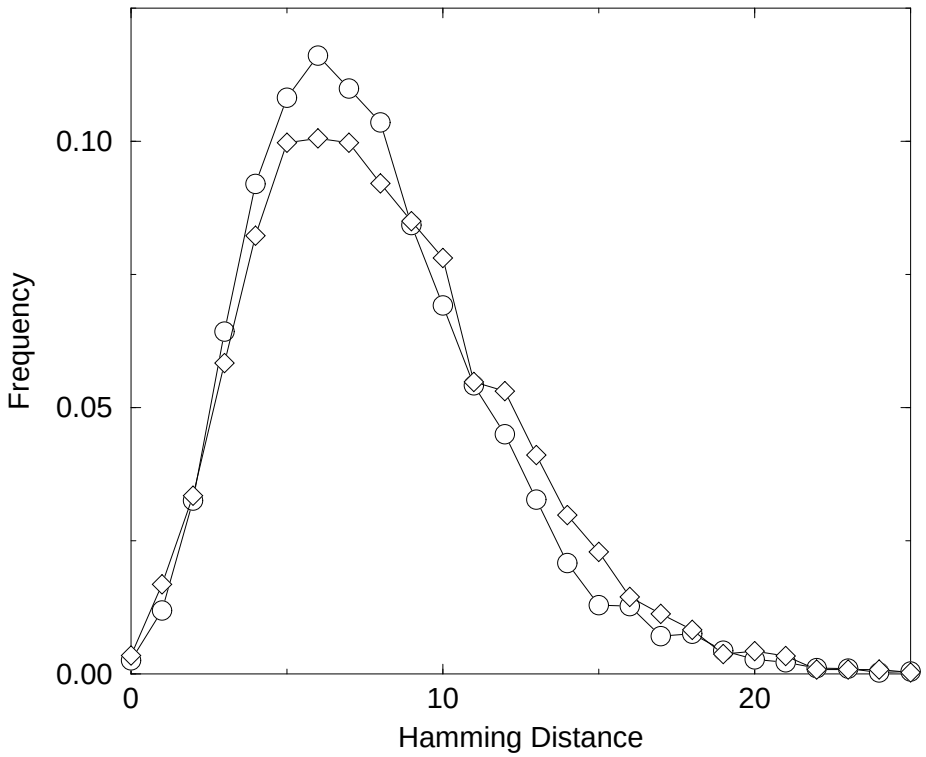


Figure 30: Distribution of Hamming distance between starting and solution sequence from the inverse fold algorithm. Data for the **AUGC** alphabet are shown with circles ○, diamonds ◇ are used for **GC**. 100 target structures derived by folding random sequences were used, for each target structure the inverse fold algorithm was called 200 times using different start sequences compatible with the target structure. A total of 8227 and 7539 sequences were found in the **AUGC** and **GC** case, respectively.

Some information can already be gotten from the performance of the inverse fold algorithm shown in (figures 30, 31). At a chain length of 100 a successful inverse fold on average finds a solution in a Hamming distance of about 7 from a starting sequence compatible with the target structure in the **AUGC** alphabet.

Clearly compatibility to the target structure is a prerequisite for any solution sequence. For a target structure with b base pairs and an alphabet with stickiness p we have to exchange on average $(1-p)b$ bases to obtain a compatible sequence, since $(1-p)$ is the probability that a sequence cannot form a particular pair and provided every base in the alphabet can form a pair with at least one other base. **AUGC** sequences of length 100 form about

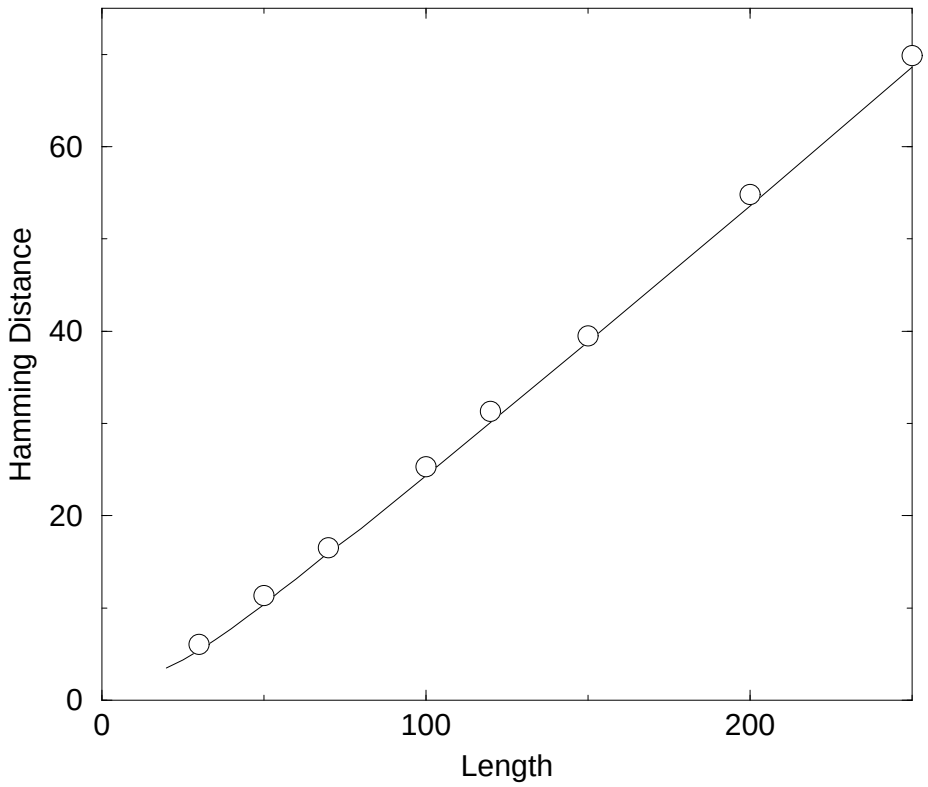


Figure 31: Mean distance from random start sequence to solution and mean number of base pairs $\bar{b}$ as a function of sequence length n. 100 target structures from the **AUGC** alphabet were used at each length. A Hamming distance of $\frac{5}{8}\bar{b}$ is needed on average to find a compatible sequence.

24.4 pairs on average, therefore on average $24.4 \cdot 5/8 = 15.2$ point mutations are needed to go from a random starting sequence to a sequence compatible with some target structure. Note that this is more than twice the distance added by the inverse folding. For **GC** sequences of length 100 we typically need $37.1 \cdot 1/2 = 18.5$ mutations to make a sequence compatible versus a distance of 8 added by the inverse folding, for **AU** sequences the numbers are $19.1 \cdot 1/2 = 14.5$ and 6 respectively.

To obtain a more reliable estimate for the shape space covering radius, we designed the following computer experiment: A target sequence is chosen at random and a reference structure is generated by folding another random structure. A first trial sequence folding into the reference structure is then generated by inverse folding starting from the target sequence. We then

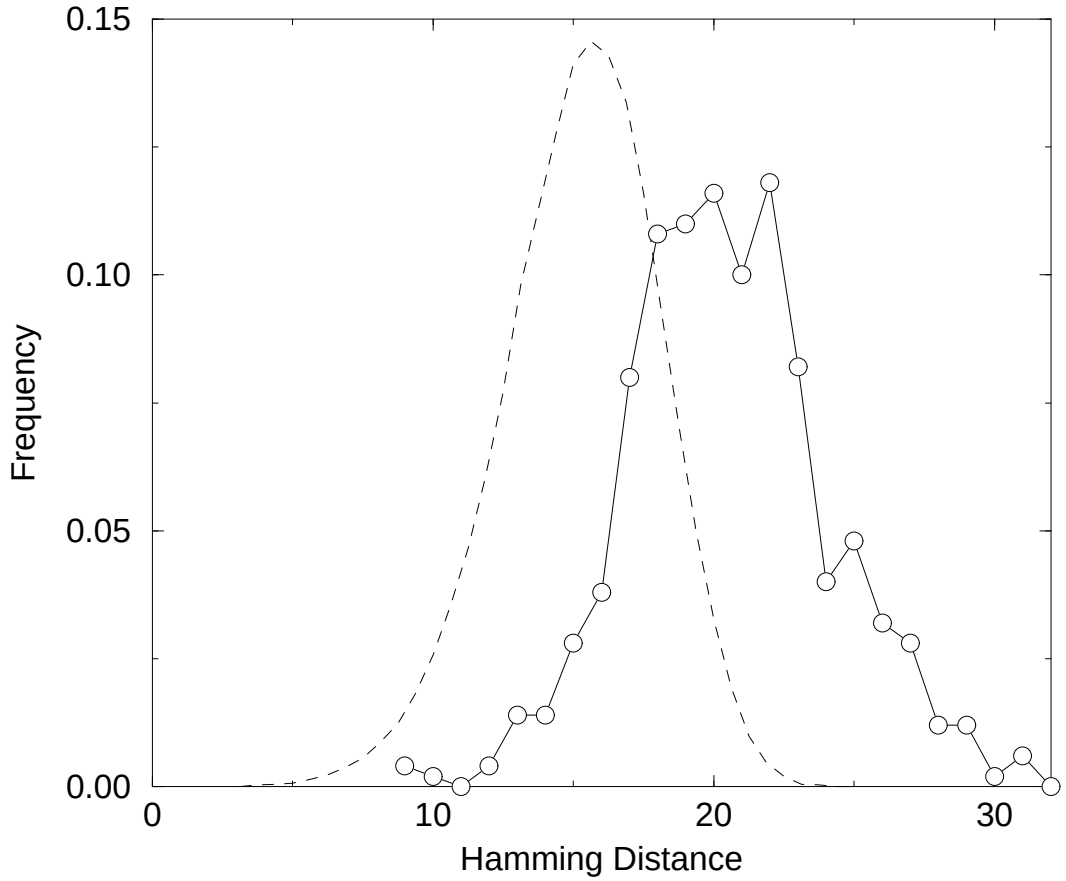


Figure 32: Nearest distances from a target sequence at which a reference structure could be found for 500 pairs of sequences and structures form the **AUGC** alphabet with chain length n=100. For each pair the best result from 100 trials as described in the text is shown. The dashed line is the corresponding distribution of base pair frequencies with the x values scaled by $1-p$=5/8. This distribution gives a lower bound on the distance where the reference structure can be found while the curve to the right provides an upper bound.

search for neighbors of the trial sequence that fold into the reference structure but lie closer to the target using point mutations on unpaired bases or exchange of two bases against another possible pair for paired ones. If such a sequence is found, it is accepted as the new trial sequence, and the procedure is repeated until no further approach to the target is possible. The Hamming distance of the endpoint of this adaptive walk to the target sequence recorded and the procedure is repeated starting from the inverse fold. The distances thus recorded are upper bounds to the minimal distance from the target in

Table 9. Lower and upper bounds for shape space covering radius.

n	**AUGC**		**GC**		**AU**	
50	6.5	9.2	8.5	10.7	6.0	7.0
70	10.0	13.7	12.5	15.6	9.3	11.5
100	15.2	20.5	18.6	22.9	14.6	17.3

Remark. The upper bounds shown here are mean values from 200 pairs of structure and target sequence for length 50, 500 pairs for length 100 **AUGC** sequences and 100 pairs otherwise. The lower bounds are derived from the mean number of base pairs, see text.

which the reference structure can be found.

The resulting distribution of distances is shown in figure 32 for **AUGC** sequences of length 100. Alphabet and chain length dependence of the mean values are shown in table 9. Obviously, the covering radius is dominated by the number of mutations necessary to find a compatible sequence. On the other hand the upper bounds could still be sharpened if more trials could be performed per structure sequence pair (i.e. if computers were faster).

In the **AUGC** alphabet, allowing **G-U** pairs, there on average 3 possible mutations to make a sequence compatible with a particular base pair. This means, if m mutations are needed to make some sequence compatible to a reference structure, we can choose from about 3^m compatible sequences in the minimal distance m. For chain length 100 this yields typically $3^{15.2} = 1.8 \cdot 10^7$ compatible sequences. It seems likely that at least one of these would fold into the reference structure, that is, the lower bound given above may actually be sharp and the number of point mutations needed to find a given structure is just the number of mutations needed to find a compatible sequence.

In any case the covering radius h_c is much smaller than the diameter of the sequence space. For **AUGC** sequences of chain length 100 a ball of radius 16 contains about 6×10^{25} sequences. Although this number is large, it is nothing compared to the total number of sequences: $4^{100} \approx 1.6 \times 10^{60}$.

A related but different question is given two structures $\mathcal{S}_1$ and $\mathcal{S}_2$ what is the minimum Hamming distance of two sequences that fold into the two structures, respectively. An upper bound to this distance of closest approach can be calculated in a variation of the experiment described above: We

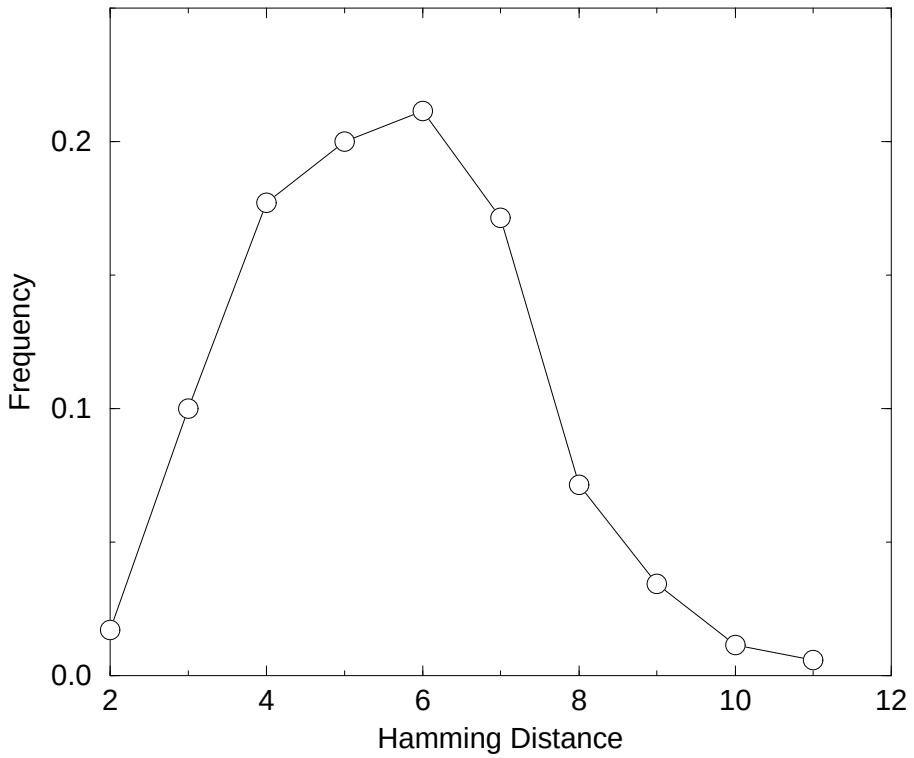


Figure 33: Distribution of upper bounds on the distance of closest approach for 350 pairs of structures of length 100 from the **AUGC** alphabet.

construct a first sequence I_1 folding into $_1$, then try to find a sequence I_2 folding into $\mathcal{S}_2$ as near as possible to I_1 as in the experiment before. We then keep I_2 fixed and try to minimize the distance by mutating I_1 without changing its structure. The process is then iterated until the distance cannot be further decreased.

Since one can always find a sequence that is compatible to both $\mathcal{S}_1$ and $\mathcal{S}_2$ the lower bound for this distance is 1.

4.6. Neutral Networks

A characteristic feature of the RNA folding map is its high neutrality. In the **AUGC** alphabet about every third point mutation will not change the structure. This is an important distinction from combinatory landscapes such as spin glasses, traveling salesman problem etc.

Table 10. Mean values for the upper bound on the distance of closest approach as a function of alphabet and chain length.

n	**AUGC**	**GC**	**AU**
50	2.1	5.6	2.6
70	3.4	9.3	4.6
100	5.6	13.0	7.8

Remark. Data are averaged over 300 pairs of structures for the **AUGC** alphabet, otherwise 100 pairs were used.

As we have seen before the sequences folding into the same secondary structure S are randomly distributed in sequence space. Because of the high probability for finding neutral neighbors these sequences are not isolated, but form clusters in sequences space. Hence, the question arises how far such sets of neutral sequences extend. This can be done in the following computer experiment. Starting from a random initial sequence I_0 we construct a monotonously diverging "neutral path" by mutating our test sequence I_n, accepting the mutated sequence I_{n+1} if the mutation is neutral $\mathcal{S}(I) = \mathcal{S}(I_0)$ and the Hamming distance does not decrease $d(I_(n+1), I_0) \geq d(I_n, I_0)$. To make sure the procedure will halt eventually, we allow at most 10 steps where $d(I_(n+1), I_0) = d(I_n, I_0)$ in a row. As mutations we again allow the exchange of a single unpaired base or the exchange two bases paired in the reference structure.

The length $\mathcal{L}$ of a path is the Hamming distance between the reference sequence and the last sequence, and hence a lower bound on the diameter of the connected "neutral network". Clearly, a neutral path cannot be longer than the chain length, $\mathcal{L} \leq n$. The length distribution of neutral paths in the sequence space of RNA molecules of chain length $n = 100$ is shown in figure 34 for the the natural **AUGC** and **GC** alphabet.

It is remarkable that more than 20 % of the neutral paths in the sequence space of natural **AUGC** sequences have the maximum length. They lead

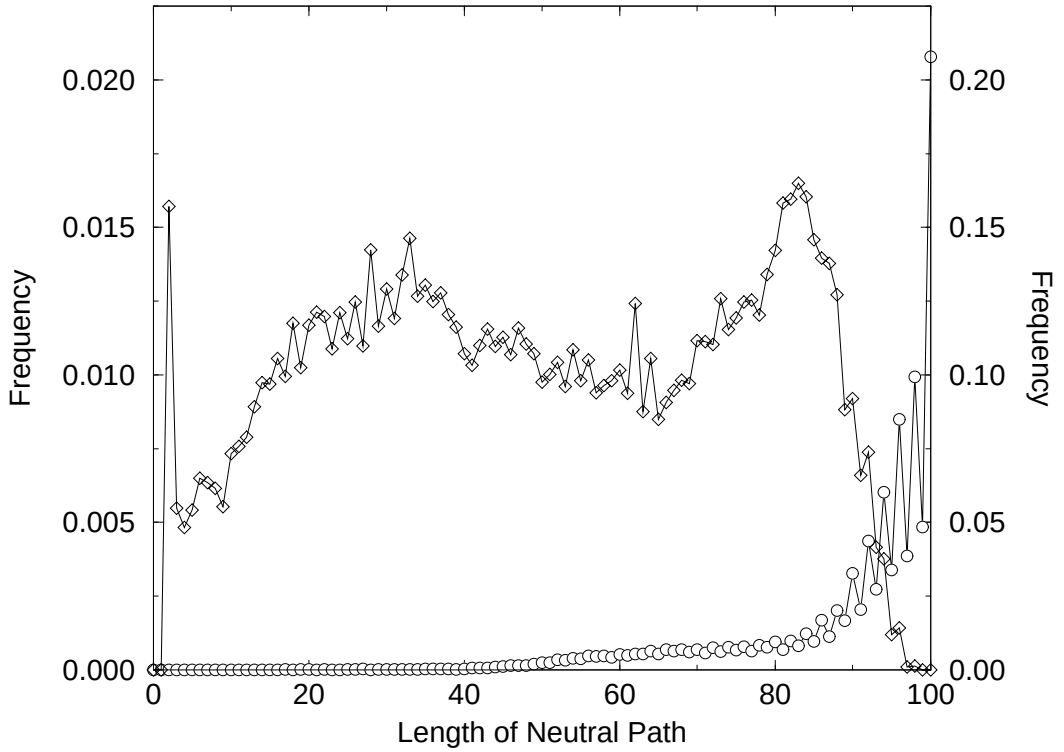


Figure 34: Lengths of Neutral Paths. Longest distances between the reference and the end points of monotonously diverging neutral paths for the **AUGC** ○ (right scale) and **GC** ◇ (left scale) alphabets.

through the entire sequence space to a sequence differing in all positions from the reference but still sharing its structure. In shape spaces derived from binary sequences almost no neutral path reaches the complementary sequence. This is partly a consequence of the symmetry of the binomial distribution: there are very few sequences in the error classes $n-1$, $n-2$, etc., and it is unlikely that we find one among them which folds precisely into the same structure as the reference sequence. Still the average length $\mathcal{L}$ is much larger than the average distance of two randomly chosen sequences, $\mathcal{L} \gg n/2$.

The union of all neutral paths forms a dense neutral network in the example considered here. This, of course, need not be the case in general: rare structures may have short neutral paths confined to small disjoint regions in sequence space. Nevertheless, neutral nets are not a peculiarity of the few most frequent structures. As shown in figure 35 even the rarest structures

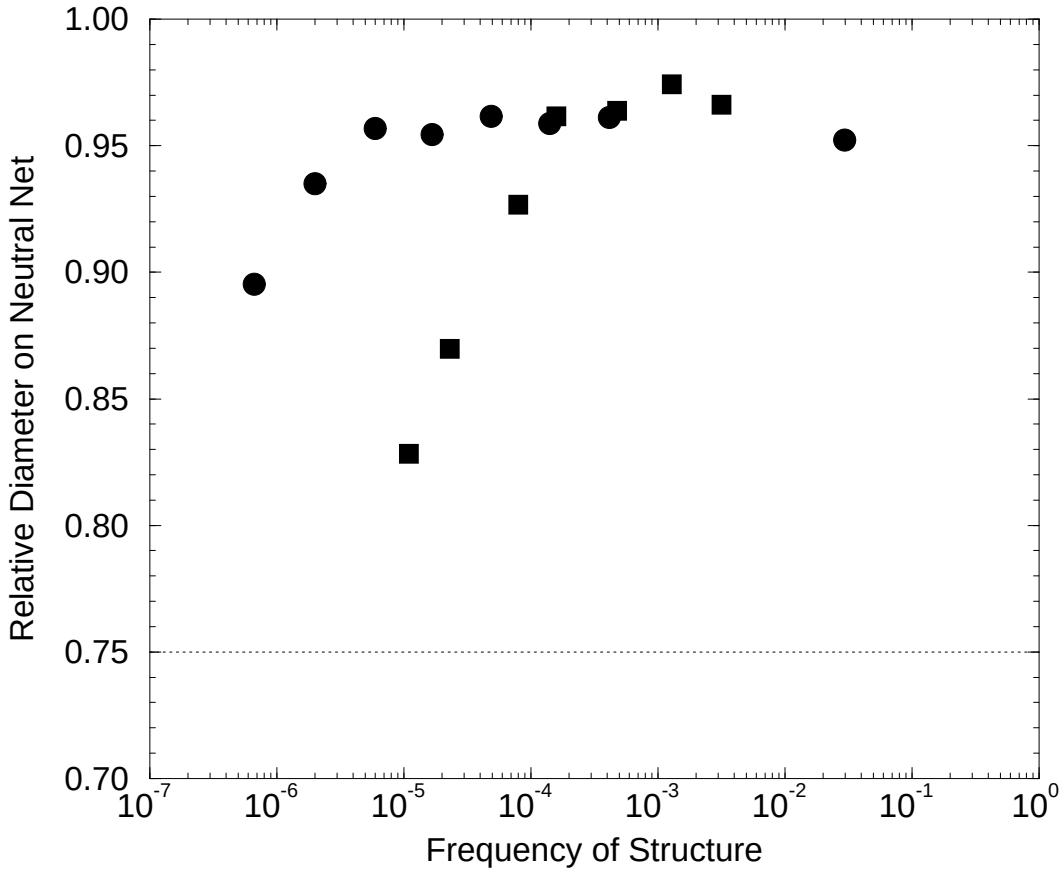


Figure 35: Average length of neutral paths as a function of the frequency of the underlying structure for **AUGC** sequences of length ◻ n=30 and • n=40. The data shown here are lower bounds on the average diameter of the neutral networks.

we were able to find give rise to networks that reach way beyond the average distance of random sequences. The data indicate, however, a sharp (chain length dependent) transition at which the average diameter of neutral networks begins to decrease with decreasing frequency of the underlying secondary structure. Such a situation is reminiscent of percolation problems in physical systems. This percolation transition has recently been studied using an abstract model based on random graphs in which a neutral net is constructed by randomly connecting neighboring points in sequence space such obtain a given average number of neutral neighbors [89]. If the frequency of neutral mutations is above 0.37 for unpaired bases and above 0.3 for pair mutations the model suggests that the resulting neutral net will span the whole sequence space. Furthermore two such neutral nets will come close

to each other within a Hamming distance of 2. Note that the frequency of neutral mutations (0.49 and 0.45 for unpaired bases and pair mutations) are way above these thresholds.

From the existence of such neutral networks one can expect far reaching consequences for evolutionary optimization where the fitness depends structure: Given a suitable error frequency an evolving population should perform a random walk along the neutral net, until it reaches a point where a better secondary structure can be reached within a few mutations (i.e. a neutral net with higher fitness comes sufficiently near). During the times where the population diffuses on the neutral net, only the phenotype is conserved while genotypic information is unstable. For even lower error frequencies the population should localize in sequence space at a point on the neutral net where the number of neutral neighbors is especially large. Simulations to test these predictions are currently under way.

5. Conclusions

Among the most important steps in understanding evolutionary adaptation is the construction of model fitness landscapes based on a proper abstraction of the adapting entities. Since fitness is evaluated on the phenotype level the crucial step lies in modeling the mapping from genotype to phenotype. This work explored in detail the statistics of the simplest realistic and biologically motivated genotype-phenotype mapping induced by RNA folding.

While sequences are linear strings, RNA secondary structures represent planar graphs or trees. In chapter 2 we studied the statistical features of RNA secondary structures analytically from a combinatorial point of view. Recursions and first order asymptotics could be derived for the frequencies of more important structural motifs. Furthermore, this allowed us to introduce random structures as a reference point for comparison with the results of folding algorithms.

The sequence/structure relationships were then explored in a series of computer experiments. To this end known algorithms for RNA structure prediction and and for comparison of structures were implemented in an efficient and easy to use library, together with a new "inverse folding" algorithm that allows to find sequences folding into a predefined secondary structure.

The results of these numerical experiments can be summarized as follows:

(1) Sequences folding into one and the same structure are distributed randomly in sequence space,
(2) The frequency distribution of structures is sharply peaked (there are comparatively few common structures and many rare ones),
(3) Sequences folding into a predefined common structure are found within (relatively) small neighbourhoods of any random sequence, and
(4) The shape space contains extended "neutral networks" joining sequences with identical structures, often these networks span the entire diameter of the sequence space.

These data show that optimization of structures by evolutionary trial and error strategies is much simpler than often assumed. In fact, the combinatory map of RNA secondary structures is *ideally* suited for evolutionary

adaptation. Exploration is easy because of vast neutral networks, and optimization is feasable since a sequence with desired secondary structure is typically few mutations away from almost anywhere in sequence space. We expect that populations that replicate with sufficiently high error rates will readily spread along these networks and to distant regions in sequence space. Other networks will pass by close enough to allow the transition to other better structures, leading to sudden step-like increases in fitness. A reduced mutation rate should cause the population to condense in the most favorable part of the neutral network and to adapt locally in this region [7, 90].

It is worth noting that these properties hold only for the biophysical **AUGC** alphabet. The binary **GC** alphabet exhibits a slightly larger covering radius, a more rugged landscape as seen in the SDS and correlation length analysis [34], and most importantly it exhibits much shorter neutral paths.

Our results are relevant for natural selection as well as for artificial selection in biotechnology. We predict that there is no need to systematically search huge portions of the sequence space, nor does one need specially designed initial conditions. These properties provide further support for the widespread applicability of molecular evolution [91, 92, 4].

6. References

[1] Charles Darwin. *The Origin of Species*. reprinted in Penguin Classics, 1859.

[2] Sewall Wright. The roles of mutation, inbreeding, crossbreeeding and selection in evolution. In D. F. Jones, editor, *int. Proceedings of the Sixth International Congress on Genetics*, volume 1, pages 356–366, 1932.

[3] Manfred Eigen and Peter Schuster. The hypercycle A: A principle of natural self-organization: Emergence of the hypercycle. *Naturwissenschaften*, 64:541–565, 1977.

[4] M. Eigen, J. McCaskill, and P. Schuster. The molecular Quasispecies. *Adv. Chem. Phys.*, 75:149 – 263, 1989.

[5] S. A. Kauffman. Adaption on rugged fittness landscapes. In D. Stein, editor, *Complex Systems*, pages 527–618. Addison Wesley, Redwood City (Cal.), 1989.

[6] A. S. Kauffman and S. Levin. Towards a general theory of adaptive walks on rugged landscapes. *J. Theor. Biol.*, 128:11–45, 1987.

[7] W. Fontana and P. Schuster. A computer model of evolutionary optimization. *Biophysical Chemistry*, 26:123–147, 1987.

[8] Catherine A. Macken and Alan S. Perelson. Protein evolution on rugged landscapes. *Proc. Natl. Acad. Sci. USA*, 86:6191–6195, 1989.

[9] C. A. Macken, P. S. Hagan, and A.S. Perelson. Evolutionary walks on rugged landscapes. *SIAM J. Appl. Math.*, 51:799–827, 1991.

[10] J. H. Gillespie. Molecular evolution over the mutational landscape. *Evolution*, 38:1116–1129, 1984.

[11] S. A. Kauffman and S. Johnsen. Co-evolution to the edge of chaos: Coupled fitness landscapes, poised states and co-evolutionary avalanches. In C. Taylor J.D. Farmer C. Langton and S. Rasmussen, editors, *Artificial Life II*, pages 325–369. Adison Wesley, Redwood City, 1991.

[12] D. R. Mills, R. L. Peterson, and S. Spiegelman. An extracellular darwinian experiment with a self-duplicating nucleic acid molecule. *Proc. Nat. Acad. Sci., USA*, 58:217–224, 1967.

[13] S. Spiegelman. An approach to the experimental analysis of precellular evolution. *Quart. Rev. Biophys.*, 17:213, 1971.

[14] F. R. Kramer, D. R. Mills, P. E. Cole, T. Nishihara, and S. Spiegelman. Evolution *in vitro*: sequence and phenotype of a mutant RNA resistant to ethidium bromide. *J. Mol. Biol.*, 89:719–736, 1974.

[15] C. K. Biebricher. Darwinian selection of self-replicating RNA molecules. *Evolutionary Biology*, 16:1–52, 1983.

[16] C. K. Biebricher, M. Eigen, and W. C. Gardiner Jr. Kinetics of RNA replication. *Biochemistry*, 22:2544–2559, 1983.

[17] C. K. Biebricher, M. Eigen, and W. C. Gardiner Jr. Kinetics of RNA replication: Competition and selection among self-replicating RNA species. *Biochemistry*, 24:6550–6560, 1985.

[18] G. Strunk. *Automatized evolution experiments in vitro and natural selection under controlled conditions by means of the serial transfer technique*. PhD thesis, Universität Braunschweig, 1993.

[19] Tom Cech. RNA as an enzyme. *Scientific American*, 11:76–84, 1986.

[20] C. Guerrier-Takada, K. Gardiner, T. Marsh, N. Pace, and S. Altman. The RNA moiety of ribonuclease P is the catalytic subunit of the enzyme. *Cell*, 35:849–857, 1983.

[21] C. Guerrier-Takada and S. Altman. Catalytic activity of an RNA molecule prepared by transcription *in vitro*. *Science*, 223:285–286, 1984.

[22] W. Gilbert. The RNA world. *Nature*, 319:618, 1986.

[23] Gerald. F. Joyce. The rise and fall of the RNA world. *The New Biologist*, 3:399–407, 1991.

[24] M.S.Z. Horwitz, D.K. Dube, and L.A. Loeb. Selection of new biological activities from random nucleotide sequences: evolutionary and practical considerations. *Genome*, 31:112–117, 1989.

[25] A.D. Ellington and J.W. Szostak. In vitro selection of RNA molecules that bind specific ligands. *Nature*, 346:818–822, 1990.

[26] D. P. Bartel and J. W. Szostak. Isolation of new ribozymes from a large pool of random sequences. *Science*, 261(5127):1411, 1993.

[27] Amber A. Beaudry and Gerald F. Joyce. Directed evolution of an RNA-enzymee. *Science*, 257:635–641, 1992.

[28] Niles Lehman and Gerald F. Joyce. Evolution in vitro of an RNA enzyme with altered metal dependence. *Nature*, 361:182–185, 1993.

[29] Francois Major, Marcel Turcotte, Daniel Gautheret, Guy Lapalme, Eric Fillion, and Robert Cedergren. The combination of symbolic and numerical computation for three-dimensional modeling of RNA. *Science*, 253:1255–1260, 1991.

[30] E. Westhof. Modelling the three - dimensional structuer of ribonucleic acids. *J. of Molecular Structure (Theochem)*, 286:203 – 210, 1993.

[31] A. S. Perelson and G. Oster. Theoretical studies of clonal selection: Minimal antibody repertoire size and reliability of self/non-self discrimination. *J. Theor. Biol.*, 81:645–670, 1979.

[32] L. A. Segel and A. P. Perelson. Computations in shape space: a new approach to immune network theory. In *Theoretical Immunology. Part Two*, pages 321 – 343. Addison-Wesley, Redwood City (Cal.), 1988.

[33] R. W. Hamming. In *Coding and Information Theory*, pages 44 – 47. Prentice Hall, Englewood Cliffs (N. J.), 1980,1986.

[34] Walter Fontana, Peter F. Stadler, Erich G. Bornberg-Bauer, Thomas Griesmacher, Ivo L. Hofacker, Manfred Tacker, Pedro Tarazona, Edward D. Weinberger, and Peter Schuster. RNA folding and combinatory landscapes. *Phys. Rev. E*, 47(3):2083 – 2099, 1993.

[35] C.W.A. Pleij and L. Bosch. *RNA Pseudoknots: Structure, Detection and Prediction*, pages 289 – 303. Academic Press, Methods in Enzymology, methods in enzymology 180 edition, 1989.

[36] M. S. Waterman. Secondary structure of single - stranded nucleic acids. *Studies on foundations and combinatorics, Advances in mathematics supplementary studies, Academic Press N.Y.*, 1:167 – 212, 1978.

[37] Pauline Hogeweg and B. Hesper. Energy directed folding of RNA sequences. *Nucl. Acid. Res.*, 12:67–74, 1984.

[38] D.A.M. Konings and P. Hogeweg. Pattern analysis of RNA secondary structure, similarity and consensus of minimal-energy folding. *J. Mol. Biol.*, 207:597–614, 1989.

[39] D.A.M. Konings. Pattern analysis of RNA secondary structures. *Proefschrift, Rijksuniversiteit te Utrecht*, 1989.

[40] W. Fontana, T. Griesmacher, W. Schnabl, P.F. Stadler, and P. Schuster. Statistics of landscapes based on free energies, replication and degredation rate constants of RNA secondary structures. *Monatshefte der Chemie*, 122:795–819, 1991.

[41] M. Zuker and D. Sankoff. RNA secondary structures and their prediction. *Bulletin of Mathematical Biology*, 46(4):591–621, 1984.

[42] Bruce A. Shapiro. An algorithm for comparing multiple RNA secondary stuctures. *CABIOS*, 4(3):387–393, 1988.

[43] R. Donaghey and L.W. Shapiro. Motzkin numbers. *J. Comb. Theor. A*, 23:291–301, 1977.

[44] P. R. Stein and M. S. Waterman. On some new sequences generalizing the Catalan and Motzkin numbers. *Discrete Mathematics*, 26:261–272, 1978.

[45] K. Ohmori and E. Tanaka. A unified view on tree metrics. In G. Ferrate, editor, *Syntactic and Structural Pattern Recognition*, pages 85–100, Berlin, Heidelberg, 1988. Springer-Verlag.

[46] Bruce A. Shapiro and Khaizhong Zhang. Comparing multiple RNA secondary structures using tree comparisons. *CABIOS*, 6:309–318, 1990.

[47] J. A. Howell, T. F. Smith, and M. S. Waterman. Computation of generating functions for biological molecules. *SIAM J. Appl. Math.*, 39:119–133, 1980.

[48] W. R. Schmitt and M. S. Waterman. Plane trees and RNA secondary structure. Preprint, 1992.

[49] G. Szegö. Orthogonal polynomials. In *Amer. Math. Soc. Coll. Publ.*, volume XXIII. Amer. Math. Soc., New York, 1959.

[50] E. A. Bender. Asymptotic methods in enumeration. *SIAM Review*, 16:485–515, 1974.

[51] M. S. Waterman. Combinatorics of RNA hairpins and cloverleaves. *Studies Appl. Math.*, 60:91–96, 1978.

[52] Arthur M. Lesk. A combinatorial study of the effects of admitting non-watson-crick base pairings and of base compositions on the helix-forming potential of polynucleotides of random sequences. *J. Theor. Biol.*, 44:7–17, 1974.

[53] Ruth Nussinov, George Piecznik, Jerrold R. Griggs, and Daniel J. Kleitman. Algorithms for loop matching. *SIAM J. Appl. Math.*, 35(1):68–82, 1978.

[54] R.R. Gutell. Comparative studies of RNA: Inferring higher order structure from patterns of sequence variation. *Current Opinion in Structural Biology*, 3:313, 1993.

[55] C.R. Woese, L.J. Magrum, R. Gupta, R.B. Siegel, D.A. Stahl, J. Kop, N. Crawford, J. Brosius, R. Gutell, J.J. Hogan, and H.F. Noller. Secondary structure model of bacterial 16S ribosomal RNA: Phylogenetic, enzymatic and chemical evidence. *Nucl.Acid.Res*, 8:2275–2293, 1980.

[56] R.R. Gutell. Evolutionary characteristics of 16S and 23S rRNA sequences. In H. Hartman and K. Matsuno, editors, *The Origin and Evolution of the Cell*. World Scientific, Singapore, 1992.

[57] H. M. Martinez. An RNA folding rule. *Nucl. Acid. Res.*, 12:323–335, 1984.

[58] Jan Pieter Abrahams, Mijam van den Berg, Eke van Batenburg, and Cornelis Pleij. Prediction of RNA secondary structure, including pseudoknotting, by computer simulation. *Nulc. Acid. Res.*, 18:3035–3044, 1990.

[59] A. P. Gultyaev. The computer simulation of RNA folding involving pseudoknot formation. *Nucl. Acid. Res.*, 19(9):2489 – 2494, 1991.

[60] A. A. Mironov, L. P. Dyakonova, and A. E. Kister. A kinetic approach to the prediction of RNA secondary structures. *Journal of Biomolecular Structure and Dynamics*, 2(5):953, 1985.

[61] A. A. Mironov and A. E. Kister. RNA secondary structure formation during transcription. *J. of Biomolecular Structure and Dynamics*, 4:1–9, 1986.

[62] Manfred Tacker, Peter F. Stadler, Erich G. Bornberg-Bauer, Ivo L. Hofacker, and Peter Schuster. Robust properties of RNA secondary structure folding algorithms. In preparation, 1994.

[63] Manfred Tacker. *Robust Properties of RNA Secondary Structure Folding Algorithms*. PhD thesis, University of Vienna, 1993.

[64] J. Ninio. Prediction of pairing schemes in RNA molecules - loop contributions and energy of wobble and non wobble pairs. *Biochimie*, 61:1133, 1979.

[65] W. Salser. Globin messenger RNA sequences - analysis of base-pairing and evolutionary implications. *Cold Spring Harbour Symp. Quant. Biol.*, 42:985, 1977.

[66] Susan M. Freier, Ryszard Kierzek, John A. Jaeger, Naoki Sugimoto, Marvin H. Caruthers, Thomas Neilson, and Douglas H. Turner. Improved free-energy parameters for predictions of RNA duplex stability. *Proc. Natl. Acad. Sci., USA*, 83:9373–9377, 1986.

[67] D. H. Turner, N. Sugimoto, and S. Freier. RNA structure prediction. *Annual Review of Biophysics and Biophysical Chemistry*, 17:167–192, 1988.

[68] L. He, R. Kierzek, J. SantaLucia, A.E. Walter, and D.H. Turner. Nearest-neighbour parameters for G-U mismatches. *Biochemistry*, 30:11124, 1991.

[69] John A. Jaeger, Douglas H. Turner, and Michael Zuker. Improved predictions of secondary structures for RNA. *Proc. Natl. Acad. Sci., USA, Biochemistry*, 86:7706–7710, 1989.

[70] C. Papanicolau, M. Gouy, and J. Ninio. An energy model that predicts the correct folding of the tRNA and the 5S RNA molecules. *Nucl. Acid. Res.*, 12:31–44, 1984.

[71] Vincent P. Antao and Jr. Ignacio Tinoco. Thermodynamic parameters for loop formation in RNA and DNA hairpin tetralooops. *Nucl.Acid.Res*, 20(4):819–824, 1992.

[72] M. S. Waterman and T. F. Smith. RNA secondary structure: A complete mathematical analysis. *Mathematical Biosciences*, 42:257–266, 1978.

[73] Ruth Nussinov and Ann B. Jacobson. Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proc. Natl. Acad. Sci. USA*, 77(11):6309–6313, 1980.

[74] M. Zuker and P. Stiegler. Optimal computer folding of larger RNA sequences using thermodynamics and auxiliary information. *Nulc. Acid. Res.*, 9:133–148, 1981.

[75] M. Zuker. On finding all suboptimal foldings of an RNA molecule. *Science*, 244:48–52, 1898.

[76] John S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29:1105–1119, 1990.

[77] Kaizhi Yue and Ken A. Dill. Inverse protein folding problem: Designing polymer sequences. *Proc. Natl. Acad. Sci. USA*, 89:4163–4167, 1992.

[78] M. Zuker, J. A. Jaeger, and D. H. Turner. A comparison of optimal and suboptimal RNA secondary structures predicted by free energy minimization with structures determined by phylogenetic comparison. *Nucl. Acid. Res*, 19(10):2707–2714, 1991.

[79] H. J. Bandelt and A. W.M. Dress. A canonical decomposition theory for metrics on a finite set. *Advances in mathematics*, 92(1):47, 1992.

[80] Manfred Eigen, Ruthild Oswatitsch-Winkler, and Andreas Dress. Statistical geometry in sequence space: A method of quantitative comparative sequence analysis. *Proc. Natl. Acad. Sci., USA*, 85:5913–5917, 1988.

[81] Benoit B. Mandelbrot. *The Fractal Geometry of Nature.* Freeman & Co., New York, 1983.

[82] Benoit B. Mandelbrot. An information theory of the statistical structure of language. In *Proceedings of the Symposium on Applications of Communications Theory*, London, 1953. Butterworths.

[83] G.K. Zipf. *Human Behaviour and the Principle of Least Effort.* Addison-Wesley, Reading(Mass.), 1949.

[84] Hirotoyo Ishii. The distribution of duplicate books in university libraries and its relationship to Zipf's Law. *Toshokan Gakki nenpo (Annals of Japan Society)*, 36(3):97, 1990.

[85] Wentian Li. Random texts exhibit Zipf's-law-like word frequency distribution. Technical Report 91-03-016, Santa Fe Institute, 1991.

[86] Y.S. Chen. Zipf's law in text modeling. *Int. J. General Systems*, 15:232, 1989.

[87] Katja Nieselt-Struwe. *Konfigurationsanalysen kombinatorischer und biologischer Optimierungsprobleme.* PhD thesis, Universität Bielefeld, 1992.

[88] Martin A. Huynen, Danielle A. M. Konings, and Pauline Hogeweg. Multiple coding and the evolutionary properties of RNA secondary structure. *J. Theor. Biol.*, 165(2):251, 1993.

[89] Christian Reidys, Peter F. Stadler, and Peter Schuster. Generic properties of combinatory maps and application on RNA secondary structures. Preprint, 1994.

[90] Walter Fontana, Wolfgang Schnabl, and Peter Schuster. Physical aspects of evolutionary optimization and adaption. *Physical Review A*, 40(6):3301–3321, 1989.

[91] Manfred Eigen. Selforganization of matter and the evolution of biological macromolecules. *Die Naturwissenschaften*, 10:465–523, 1971.

[92] Manfred Eigen and Peter Schuster. *The Hypercycle: a principle of natural self-organization.* Springer, Berlin, 1979 (ZBP:234.

Contents