The Tree Alignment Model: Algorithms, Implementations and Applications for the Analysis of RNA Secondary Structures

Dissertation zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat) der
Technischen Fakultät der Universität Bielefeld

vorgelegt von

Matthias Höchsmann

Bielefeld im März 2005

I think that I shall never see
A poem lovely as a tree.

A tree whose hungry mouth is prest
Against the earth's sweet flowing breast;

A tree that looks at God all day,
And lifts her leafy arms to pray;

A tree that may in Summer wear
A nest of robins in her hair;

Upon whose bosom snow has lain;
Who intimately lives with rain.

Poems are made by fools like me,
But only God can make a tree.

- Joyce Kilmer, "Trees" (poem), 1914

Acknowledgments

First of all, I thank Robert Giegerich for his support during the last years. He has a good sense for the “hot topics” in Bioinformatics before they are hot. I thank Stefan Kurtz for his advice how to write scientifically (I hope I did not violate too many rules in this thesis) and how to implement algorithms efficiently. I much appreciate the cooperativeness of Peter Stadler to appraise this thesis.

I acknowledge the support of Marc Rehmsmeier who was my mentor for scientific questions of all kind. I thank Alexander Sczyrba (German pole vault champion, seniors) for a lot of proof reading and being my training mate in the gym. I thank my brother Thomas Höchsmann for doing the space and time measurements for this thesis and proof reading. I thank Thomas Fiedler, Dirk Evers, Björn Voss, Thomas Töller, Carsten Meyer, Michael Beckstette, Robert Homann, Bernd Rössler and Jens Reeder for proof reading and helpful comments.

I appreciate the financial support of the International NRW Graduate School in Bioinformatics and Genome Research. I much appreciate Klaus Prank’s efforts who also cultivated my passion for fine dining. I thank my wife Mariléne and my son Marlon for keeping me grounded. Finally, I thank my mother for all her support and patience.

Contents

Chapter 1

Introduction

1.1 RNA - A Key Player of Life

RNA (Ribonucleic Acid) is a chain molecule. It is built from nucleotides containing the bases A(denine), C(ytosine), G(uanine), and U(racil). By folding back onto itself, an RNA molecule forms structure, stabilized by forces of hydrogen bonds between certain pairs of bases (A–U, C–G, G–U), and dense stacking of neighboring base pairs.

The central role of RNA in translation of the genetic code into proteins was proposed by Watson & Crick shortly after their discovery of the three dimensional structure of DNA in the early 50's [226]. Besides ribosomal RNA and transfer RNA, RNA was thought to be messenger RNA, carrying the genetic code from inside the nucleus to the ribosomes in the cytoplasm. The central dogma of molecular biology, enunciated by Crick in 1958, stated that the flux of information from DNA to protein is a one-way; DNA is transcribed into RNA which is subsequently translated into protein[1]. This dogma was predominant for almost three decades[2], but it turned out to be an over-simplification. With the discovery of reverse transcriptase in retroviruses [5, 199], the central dogma was extended, allowing information

[1] In fact, this is the propagated interpretation of Crick's work. In [30] Crick points out that this is due to a misunderstanding of his original work.

[2] It was still taught when I was in highschool in the 90's.

to be carried from RNA to DNA. In 1986, the discovery of catalytic RNA [17], named *ribozymes*, suggested that RNA is involved more deeply in the processes of living cells. This rekindled earlier speculations about the role of RNA in the origin of life when researchers realized that they had a classic chicken-and-egg problem. Proteins cannot exist without DNA that specify their construction, and DNA cannot replicate without proteins. As a remedy, Walter Gilbert proposed the *RNA world hypothesis* [60] which is, until now, controversially discussed. He proposed that RNA molecules first catalyzed their own replication and developed a repertoire of enzymatic activities. In the next stage, RNA molecules began to synthesize proteins, which emerged as superior enzymes because their 20 side chains are more versatile than the four bases of RNA. Finally, DNA was formed by reverse transcription of RNA. DNA replaced RNA as the generic material because its double helix is a more stable and reliable storage of genetic information than is single stranded RNA. At this point, RNA was left with roles it has retained to these days, as information carrier (mRNA) and adapter in protein synthesis (tRNA) and as critical component of ribosomes (rRNAs) and other assemblies that mediate gene expression. The present intricate mechanism of information transfer from gene to protein probably began when RNA alone wrote the script, directed the action, and played all the key parts.

Gene regulation remained an important function of RNAs in cell even after proteins were invented by nature. New regulation mechanisms that involve RNA molecules were identified over the last years: A *riboswitch*, sometimes referred to as *regulon*, is a part of mRNA that directly binds a small molecule. Riboswitches are involved in regulating gene activity in response to the presence or absence of their target which could be certain molecules [230] or environmental parameters like temperature [145]. Thus, mRNA that contains a riboswitch is directly involved in regulating its own activity. Riboswitches are a demonstration that naturally occurring RNA can bind small molecules, a capability that many previously believed to be the domain of proteins. *Small nuclear RNA* (snRNA) is the name used to refer to a number of small RNA molecules found in the nucleus. These RNA molecules are important in a number of processes including RNA splicing and maintenance of telomeres, or chromosome ends [212]. *Untranslated terminal regions* (UTRs) of mRNAs sometimes contain regulatory motifs which are important for the posttranscriptional gene regulation. Such motifs can affect mRNA localization [91], mRNA degradation [69], and translational regulation [65]. The recently discovered *microRNAs* add another mechanism to the pool of known posttranscriptional gene regulation methods [16]. A comprehensive review of the *modern RNA world* is given in [43, 185]. It is clear that the investigation of known and the discovery of new non coding RNAs is a major task in modern molecular biology; without it, "the big picture" of gene regulation would be incomplete. Comparative analysis of RNA structures facilitates this research and the development of models and algorithms is an expanding field in Bioinformatics.

1.2 Motivation and Organization of this Thesis

Phylogenetic analysis of nucleotide sequences and amino acid sequences has proven to be extremely powerful in the analysis of genomes. The comparative analysis of coding regions, i.e. regions where the order of nucleotides code for proteins, has been studied extensively. But what if the signal is not sequential? As outlined in the previous section, there are numerous examples of RNA genes and motifs where the structure instead of the sequence determines the function (and for sure, there are a lot of unknown ones today). In this case the selective pressure acts on the structure, which conserves structure istead of sequence. In spite of all its success, pure sequence based comparative analysis gets to its limit when structural conservation is of interest. In this thesis, I focus on strategies to align the structure of RNA molecules.

In Chapter 2, I introduce basic terminology and outline topics that are related to structure comparison such as representation, visualization, and prediction of RNA structures. Section 2.5 is the heart of the chapter. I provide a complete in depth review of structure comparison approaches across different areas. I emphasize the properties of different models and relate different contributions.

In Chapter 3, I systematically derive dynamic programming algorithms for the calculation of the global alignment similarity of two forests. Moreover, I introduce new local similarity variants. The resulting algorithms are compact and suitable for a direct and efficient implementation in imperative programming languages.

In Chapter 4, I apply my algorithms to the comparison of RNA secondary structure forests. I introduce a new forest representation for RNA secondary structures which, in conjunction with a refined forest alignment model, provides a reasonable scoring model for the evolution of RNA secondary structures. Beside a global RNA structure alignment, I introduce local variants for RNA secondary structures. I demonstrate the performance of my algorithms by providing exhaustive measurements concerning the practical runtime and memory consumption. I introduce an intuitive 2d-plot for RNA secondary structure alignments that makes the results of a structural comparison usable without requiring knowledge in abstract structure representations.

In Chapter 5, I generalize the pairwise alignment model to align multiple RNA secondary structures and provide an algorithm that calculates multiple RNA secondary structure alignments. I propose a notion of consensus structures for a family of RNA molecules, the *RNA secondary structure profiles*, and provide intuitive visualizations for them. To demonstrate the usefulness of a multiple RNA secondary structure alignment, I propose a consensus structure prediction strategy for families of RNA molecules that have low sequence homology.

In Chapter 6, I present the structural alignment tool *RNAforester*. *RNAforester* supports the computation of pairwise and multiple alignments of RNA secondary structures based on the models and algorithms presented in Chapter 4 and Chapter 5.

In Chapter 7, I demonstrate the practical impact of the Algorithms that were presented in this thesis. I present a joint work with T. Töller and R. Giegerich concerning a strategy for the detection of new regulatory motifs that, as an integral part, includes the computation of local structure alignments. I exemplify a structure prediction strategy that is based on a multiple structure alignment of thermodynamically predicted structures for families of RNA structures that have a low sequence homology.

Chapter 2

Introductory Material

2.1 Preliminaries

2.1.1 Metrics

Let M be a set. A nonnegative function $f : M \times M \to \mathbb{R}^+$ is a *metric* if the following properties hold:

$$\begin{aligned} f(x,y) = 0 \Leftrightarrow x = y && \text{(identity)} \\ f(x,y) = f(y,x) && \text{(symmetry)} \\ f(x,y) \leq f(x,z) + f(z,y) && \text{(triangle inequality)} \end{aligned}$$

If only the symmetry and the triangle inequality condition are satisfied and the weaker condition $f(x,x) = 0$ holds, function f is denoted a *pseudo-metric*.

2.1.2 Sequences

Let Σ be a finite set, the *alphabet*. The elements of Σ are *characters*. $\Sigma_{\text{RNA}} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{U}\}$ is the RNA alphabet consisting of the bases **A**denin, **C**ytosin, **G**uanin and **U**racil. *Sequences* or equivalently *strings*, or *words* are written by juxtaposition of characters. In particular, let λ denote the *empty* *character*, also referred to as the *gap character* which acts as the neutral element of the juxtaposition, i.e. $\lambda a = a\lambda = a$. The set Σ^* of *strings over* Σ is defined by

$$\Sigma^* = \bigcup_{i \geq 0} \Sigma^i,$$

where $\Sigma^0 = \{\lambda\}$ and $\Sigma^{i+1} = \{aw \mid a \in \Sigma, w \in \Sigma^i\}$. The empty sequence that contains no characters or only gap characters is denoted by ε. I define the *tuple alphabet* as $\Sigma^n = \{(a_1, a_2, \ldots, a_n) \mid a_1, a_2, \ldots, a_n \in \Sigma\}$. For some $\sigma \in \Sigma^n$, σ_i identifies the ith component of σ. The symbols a, b, c, d refer to characters and $S, S_1, S_2, \ldots, S_n$ to sequences, unless stated otherwise.

The *length* of a string S, denoted by $|S|$, is the number of characters in S. I make no distinction between a character and a string of length one. If $S = uvw$ for some (possibly empty) strings u, v and w, then

- u is a *prefix* of S,
- v is a *substring* of S, and
- w is a *suffix* of S.

A prefix or suffix of S is *proper* if it is different from S. $S[i]$ is the i-th character of S. $S[i, j]$ is the substring of S beginning at $S[i]$ and ending at $S[j]$. If $i > j$, then $S[i, j]$ is the empty string.

2.1.3 Trees and Forests

Generally, a tree is an acyclic connected graph. I consider *rooted, ordered, node-labeled trees*, called *trees* for short. A distinguished node, the *root node*, imposes a partial ancestor-descendant relation on the tree nodes. Naturally, each path beginning at the root node whereas a node can be visited at most once ends in some node where it can not be further extended, a *leaf node*. A node v is a *descendant* of a node w, if v appears after w on such a path. Conversely, w is an *ancestor* of v. If v and w are directly connected by an edge, w is the *parent* of v and v is a *child* of w. Two nodes are *siblings* if they have the same parent node. The last common ancestor of v and w, denoted by $lca(v, w)$, is the node p that is an ancestor of v and w such that there is no descendant of p that satisfies the condition of being ancestor of v and w. A tree is *ordered* if the order among sibling nodes matters, i.e. there exists an order relation for each set of sibling nodes. An *ordered forest* is a sequence of trees, called *forest* for short. A function *label* assigns a character from some alphabet Σ to each node in a forest. I use $\mathcal{T}(\Sigma)$ and $\mathcal{F}(\Sigma)$ for the set of Σ-*labeled* trees and forests, respectively. The *empty tree* and the *empty graph* which contain no nodes are denoted by $\emptyset$. Where convenient, I identify a tree with the forest containing only this tree.

Since a tree is a special case of a forest, I give the following definitions in terms of forests: Let F be a forest. $V(F)$ denotes the set of nodes in F. The *size* of F, denoted by $|F|$, is the number of its nodes. The number of leaf nodes is referred to as $leaves(F)$. The length of the longest path from a root to a leaf is the *depth* of F, denoted by $depth(F)$. The *preorder index* of a node in a tree is its position in the sequence of nodes that is obtained by the following procedure: First, visit the root node. Second, apply this procedure recursively to the trees induced by the children nodes according to their left-to-right order. For forests, the preorder index is defined by the same procedure assuming a virtual root node that is not counted in the indexing. $pre_F(v)$ denotes the preorder index of node v in F.

I now give definitions of substructures in trees and forests: A *subtree* at node v of F consists of node v and all its descendants. Two subtrees are *siblings* if their root nodes are siblings. A *subforest* is a sequence of sibling subtrees. A *tree pattern* is a subtree T' whereas arbitrary subtrees of T' can be removed.

2.1.4 The Sequence Edit Distance

A fundamental model for approximate string comparison is the model of edit distance [113, 171, 213]. It measures the distance between strings in terms of edit operations, that is, deletions, insertions, and replacements of single characters. Two strings are compared by determining a sequence of edit operations that converts one string into the other and minimizes the sum of the costs of edit operations. Nowadays, the edit distance between strings is basic knowledge in computational biology and is an integral part of numerous textbooks, lectures and seminars. I give a brief introduction based on [108].

The notion of edit operations is the key to the edit distance model. I define the *alignment alphabet* Σ_λ^n as the tuple alphabet where for each of its elements at least one component is different from λ. Formally, $\Sigma_\lambda^n = (\Sigma \cup \{\lambda\})^n \setminus \{\lambda\}^n$. An *edit operation* is a pair $(\alpha, \beta) \in \Sigma_\lambda^2$. α and β are strings of length ≤ 1. An edit operation (α, β) is usually written as $\alpha \rightarrow \beta$. This reflects the operational view which considers edit operations as rewrite rules transforming a source string into a target string, step by step. In particular, there are three kinds of edit operations:

- $\alpha \rightarrow \beta$ denotes the *relabeling* of the character α by the character β,
- $\alpha \rightarrow \lambda$ denotes the *deletion* of the character α,
- $\lambda \rightarrow \beta$ denotes the *insertion* of the character β.

A relabeling $\alpha \rightarrow \beta$ where $\alpha = \beta$ is denoted a *match*. Notice that $\lambda \rightarrow \lambda$ is not an edit operation. Insertions and deletions are sometimes referred to collectively as *indels*.

Sometimes string comparison just means to measure how different strings are. Often it is additionally of interest to analyze the total difference between two strings into a collection of individual elementary differences. The most important mode of such analysis is an alignment of the strings. An *alignment* A of u and v is a sequence

$$(\alpha_1 \rightarrow \beta_1, \ldots, \alpha_h \rightarrow \beta_h)$$

of edit operations, for short *edit-sequence*, such that $u = \alpha_1 \ldots \alpha_h$ and $v = \beta_1 \ldots \beta_h$.

Note that the unique alignment of ε and ε is the empty alignment, that is, the empty sequence of edit operations. An alignment is usually written by placing the characters of the two aligned strings on different lines, with inserted dashes "–" denoting λ. In such a representation, every column represents an edit operation.

The alignment $A = (\lambda \rightarrow d, b \rightarrow b, c \rightarrow a, \lambda \rightarrow d, a \rightarrow a, c \rightarrow \lambda, d \rightarrow d)$ of the sequences $u = bcacd$ and $v = dbadad$ is written as follows:

$$\begin{pmatrix} - & b & c & - & a & c & d \\ d & b & a & d & a & - & d \end{pmatrix}$$

The notion of optimal alignment requires some scoring or optimization criterion. This is given by a cost function.

A *cost function* δ assigns to each edit operation $\alpha \rightarrow \beta$, $\alpha \neq \beta$ a positive real cost $\delta(\alpha \rightarrow \beta)$. The cost $\delta(\alpha \rightarrow \alpha)$ of an edit operation $\alpha \rightarrow \alpha$ is 0. If $\delta(\alpha \rightarrow \beta) = \delta(\beta \rightarrow \alpha)$ for all edit operations $\alpha \rightarrow \beta$ and $\beta \rightarrow \alpha$, then δ is *symmetric.* δ is extended to alignments in a straightforward way: The cost $\delta(A)$ of an alignment $A = (\alpha_1 \rightarrow \beta_1, \ldots, \alpha_h \rightarrow \beta_h)$ is the sum of the costs of the edit operations A consists of. More precisely,

$$\delta(A) = \sum_{i=1}^{h} \delta(\alpha_i \rightarrow \beta_i).$$

The *unit cost function* scores zero for matches and score one otherwise. The *edit distance* of S_1 and S_2, denoted by $\delta_{\mathrm{SE}}(S_1, S_2)$, is the minimum possible cost of an alignment of S_1 and S_2. That is,

$$\delta_{\mathrm{SE}}(S_1, S_2) = \min\{\delta(A) \mid A \text{ is an alignment of } S_1 \text{ and } S_2\}. \qquad (2.1)$$

An alignment A of S_1 and S_2 is *optimal* if $\delta(A) = \delta_{\mathrm{SE}}(S_1, S_2)$. Note that there can be more than one optimal alignment. If δ satisfies the mathematical axioms of a metric, then δ_{SE} is a metric.

2.2 Primary, Secondary and Tertiary Structure of RNA

RNA molecules can be formally described on different levels of abstraction. In messenger RNA (mRNA), coding regions of RNA molecules determine the sequence of amino acids in proteins which in turn determines the protein structure. This information, the *primary structure* of an RNA molecule, is carried as a sequence of nucleotides (bases) over the four letter alphabet $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{U}\}$. RNA molecules have the tendency to form a three dimensional conformation, the *tertiary structure*. By folding back onto itself, an RNA molecule forms structure, stabilized by the forces of hydrogen bonds between certain pairs of bases, and dense stacking of neighboring base pairs. These base-pairs G–C, A–U and G–U, in order of their strength, are denoted *canonical base-pairs*. In fact, almost every other base-pair combination could exist, and has been observed in nature, but their contribution to the stability of the molecule are minor in comparison with the canonical base-pairs. External factors like cellular RNAs and proteins do also influence the structure. Crystallographic studies by X-ray diffraction and nuclear magnetic resonance (NMR) can reveal the tertiary structure of an RNA molecule with high accuracy [89, 100]. Although great progress has been made, crystallographic studies are still time consuming and expensive. Moreover, tertiary structure eludes from efficient algorithms for structure prediction and comparison. In particular, these problems are reported to be NP-hard for tertiary structures [94, 122]. From a biological viewpoint, RNA tertiary structure is likely formed hierarchically. First, stable stems are formed and afterward tertiary interactions are built. The strength of additional tertiary interactions is thought to be too small to significantly change the secondary structure conformation [13, 152, 156, 202]. For economical, biological and computational reasons, a subset of tertiary structures, the RNA secondary structures [36, 50], draw researchers attention.

An *RNA secondary structure* (S, P) consists of a sequence $S \in \Sigma_{\text{RNA}}$ and a set of base-pairs $P = \{(i, j)\}$ such that $i, j \in [1, \dots, |S|]$ and $i < j$. For all $(i, j), (i', j') \in P$ the following holds: W.l.o.g let $i < i'$,

1. $i = i' \Leftrightarrow j = j'$, i.e. there is a one-to-one relation between paired bases.
2. and it holds either:
   (a) $i < j < i' < j'$, i.e. (i, j) *precedes* (i', j'), or
   (b) $i < i' < j' < j$, i.e. (i, j) *includes* (i', j').

(S, P) is a *tertiary structure* if Condition 1 is satisfied. Figure 2.1 shows an example of the primary, secondary, and tertiary structure of an RNA molecule.

An intermediate between secondary and tertiary structures are *pseudo-knotted structures* which consider certain kinds of tertiary interactions. This is an emerging field but nowadays there is still a lack of algorithms and Bioinformatics tools that handle pseudo-knotted structures efficiently.

2.3 Representation and Visualization of RNA Structures

Understanding the macromolecular structure of an RNA molecule and its relation to function still requires expert knowledge and intuition from biologists. Visualization of RNA structures is a preliminary for this task. The topology of an RNA molecule is relevant to classify RNA structures or to search for structurally homologous RNA molecules. This typically involves the visualization of secondary and pseudo-knotted structures. A visualization of tertiary structures, based on the relative position of atoms, obtained by NMR spectroscopy or X-Ray diffraction, can give insights into macromolecular mechanisms.

The most common and biological informative drawing of RNA secondary structures is a *2d-plot*, sometimes referred to as *squiggle-plot*. Embedded in a plane, paired bases are drawn adjacent to each other. Base-pair bonds

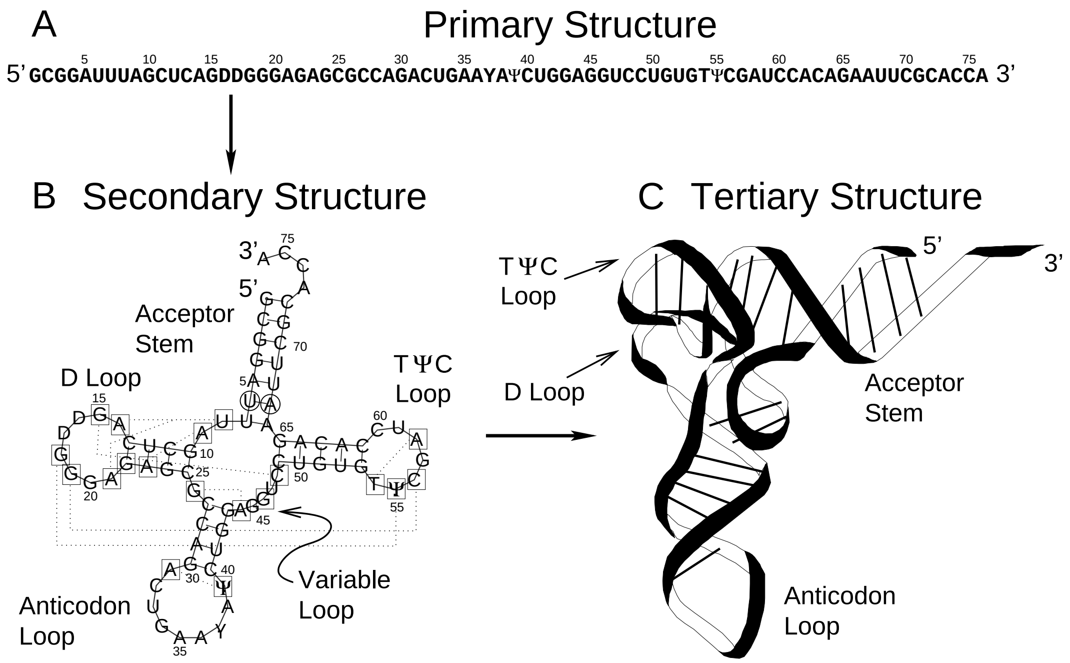


Figure 2.1: *[54] Primary, secondary and tertiary structures of yeast phenylalanine tRNA. A: The sequence was obtained from* The Genomic tRNA Database *[116, 117]. B: The secondary structure was inferred from an alignment of yeast tRNA-PHE sequences by* RNAalifold *[82], circled bases indicate neutral mutations with respect to the displayed secondary structure. Pseudo-knots and non-canonical base-pairs are indicated with a dashed line connecting squared bases [188]. C: A cartoon representation of tRNA tertiary structure, based upon tertiary structures obtained from the Protein Databank Bank (ID 6TNA,1EHZ) [99, 182].*

and the backbone of an RNA molecule are indicated as lines that do ideally not intersect. Several layout algorithms that generate 2d-plots have been proposed in [14, 109, 143, 179, 234]. The *RNAViz* [33, 34] software allows a manual fine tuning of drawings for producing publication-quality secondary structure drawings, e.g. the display of structural elements such as pseudo-knots or unformatted areas is possible. *RNA_d2* [153], *RNAdraw* [132] and *XRNA* [233] are alternative tools within this scope. Recently, a layout algorithm for pseudo-knotted structures that produces non-overlapping drawings was proposed which is implemented in the tool *Pseudoviewer* [72, 74]. The visualization of the three dimensional structure of an RNA molecule belongs to the general field of three dimensional macromolecule visualization. Beside freely distributed software like *RasMol* [175], there are many commercial tools that offer visualization of macromolecules in the framework of drug discovery.

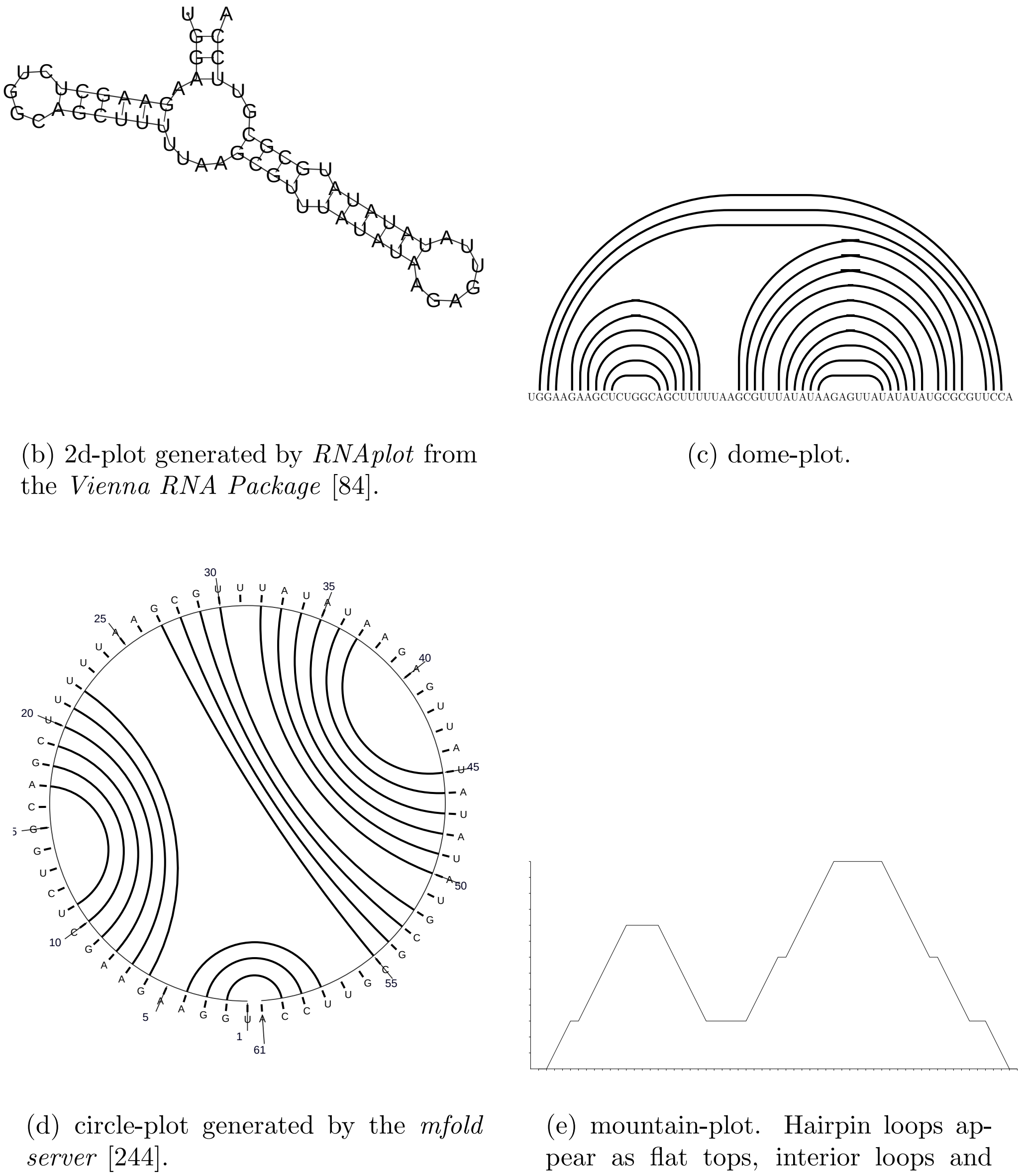
Although 2d-plots are pleasant to read, it is difficult to compare them or extract topological information. The dome-, circle- and mountain-plot address this problem. In a *dome-plot*, base-pair bonds are drawn as arcs above the sequence which is drawn as a straight line. In a *circle-plot*, the sequence is arranged as a circle and chords inside the circle connect base-pairs [151]. The *mountain-plot* draws the mountain-function of an RNA secondary structure which intuitively assigns to each nucleotide the number of base-pairs that enclose it [87]. Formally, we define the *mountain-function* for an RNA secondary structure (S, P) as follows:

$$\begin{aligned} h(0) &= 0 \\ h(i) &= \begin{cases} h(i-1)+1 & \text{if } (i,j) \in P \text{ for some } i \in [i, |S|] \\ h(i-1)-1 & \text{if } (i,j) \in P \text{ for some } j \in [1, |S|] \\ h(i-1) & \text{otherwise} \end{cases} \\ & \text{where } i > 0 \end{aligned} \tag{2.2}$$

A more technical representation are RNA secondary structure strings, for their exhaustive use in the *Vienna RNA Package* referred to as *Vienna strings* [84]. Vienna strings are sequences where, in order of the primary structure, the characters '`(`' and '`)`' denote the $5'$ and $3'$ bases of a base-pair, respectively, while '`.`' denotes an unpaired base. In addition, a second string can hold the primary structure information. Vienna strings and *Zuker-CT* files of the *mfold* software [244] are the most common formats to electronically store RNA secondary structures. In the era of web services, *RNAML* is a suggestion of a XML based standardization which is designed for the transmission of information among the RNA community [227]. An example of RNA secondary structure drawings and representations is given in Figure 2.2.

```
UGGAAGAAGCUCUGGCAGCUUUUUAAGCGUUUAUAUAAGAGUUAUAUAUAUGCGCGUUCCA
.(((.((((((.....))))))....((((.((((((.......)))))).))))..))).
```

(a) Vienna string.


UGGAAGAAGCUCUGGCAGCUUUUUAAGCGUUUAUAUAAGAGUUAUAUAUAUGCGCGUUCCA
1
5
10
20
25
30
35
40
45
50
55
61

(b) 2d-plot generated by *RNAplot* from the *Vienna RNA Package* [84].

(c) dome-plot.

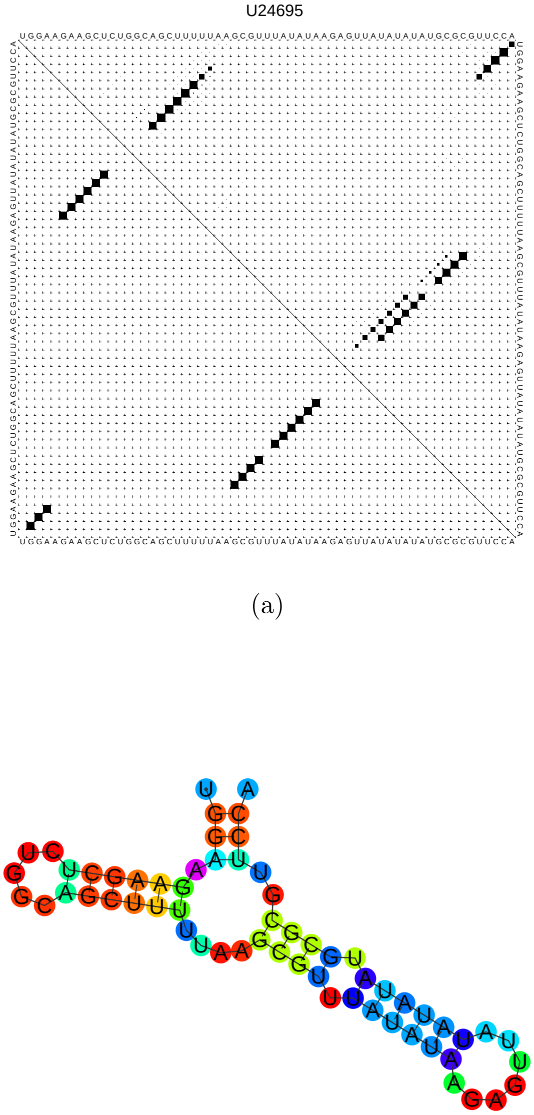
(d) circle-plot generated by the *mfold server* [244].

(e) mountain-plot. Hairpin loops appear as flat tops, interior loops and bulges as intermediate plateau, helices as sloping hillsides, and branching regions as valleys.

Figure 2.2: *Visualization of a secondary structure for the Nanos 3' UTR translation control element taken from the* Rfam *database [67] (Id: RF00161,* EMBL *Id: U24695.1).*

These visualization show a single structure of an RNA sequence. *Dot-plots* visualize the structure space of an RNA sequence with the potential to reveal suboptimal structures that are biologically relevant. Arranged in a matrix, the probabilities of base-pairs are plotted as dots whose diameter is proportional to their probability in the structure space. The base-pair frequency information has subsequently been included in single structure visualizations and likely base-pairs can be distinguished from unlikely base-pairs by a color gradient or some other indicator [245]. See Figure 2.3 for an example of a dot-plot and an annotated 2d-plot. *RNAmovies* is an interactive software for the visualization of secondary structure spaces [57]. It automatically generates animated 2d-plots where structures are morphed to explore the structure space of an RNA molecule.

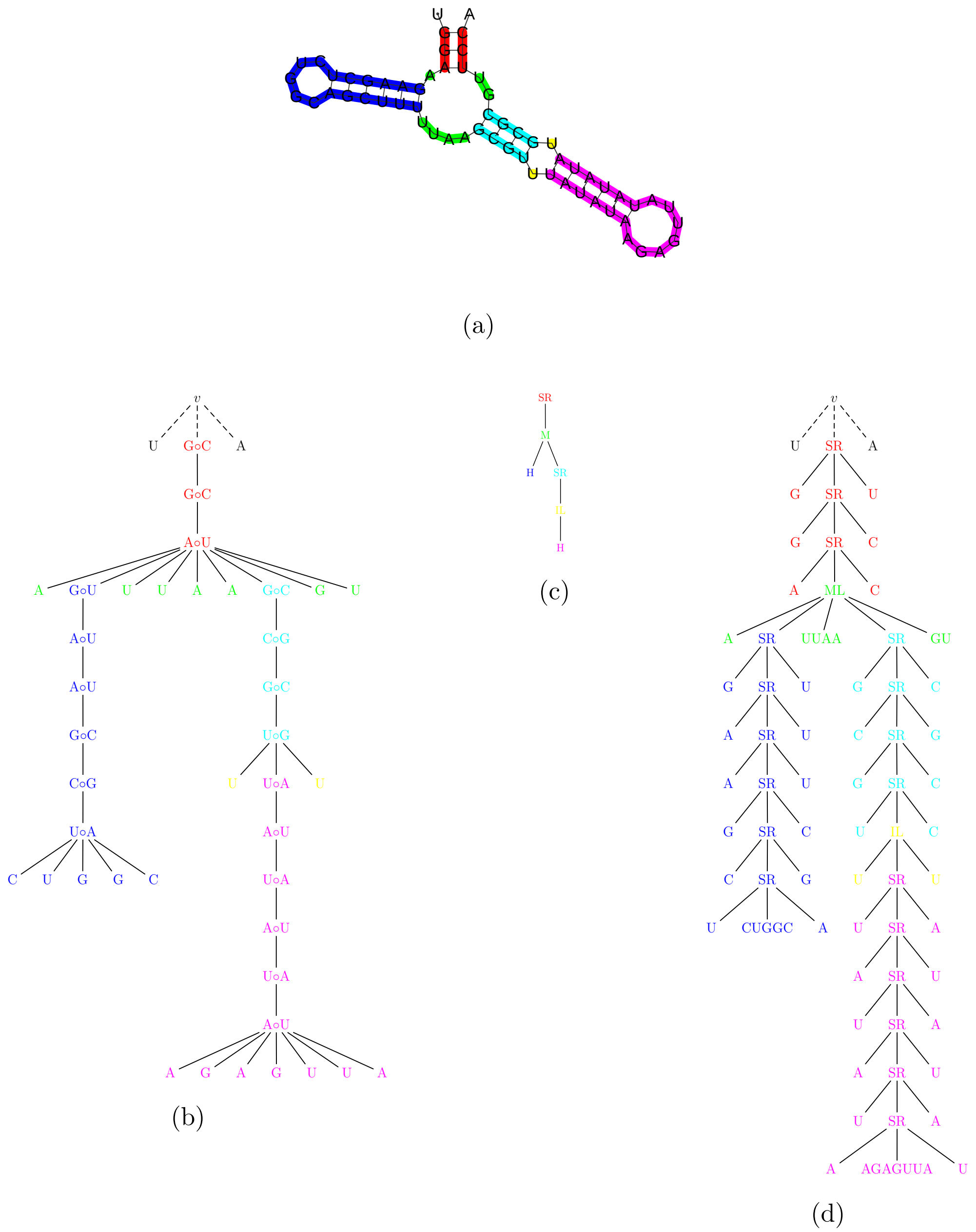
From the viewpoint of computer scientists, RNA secondary structures are often represented as trees or forests. The parent and sibling relationship of nodes is determined by the nesting of base-pair bonds. The $5'$ to $3'$ nature of an RNA molecule imposes the order among sibling nodes. This produces a forest structure in general but a virtual root node can always turn a forest into a tree. Different tree representations that vary in their resolution have been proposed. A tree structure where base-pairs correspond to internal nodes while unpaired bases correspond to leaves in the tree was proposed in [173]. I refer to it as the *natural tree representation*. A *coarse grained tree representation* where nodes correspond to the structural components - stacking regions, hairpins, bulges, internal loops and multiloops - was proposed in [110, 178, 180]. Parse trees of grammar based prediction strategies for RNA secondary structures represent the structure such that the sequence information corresponds to the preorder sequence of leaves while the internal nodes correspond to productions of the grammar [167]. An example of tree representations of RNA structures is shown in Figure 2.4.


U24695
UGGAAGAAGCUCUGGCAGCUUUUUAAGCGUUUAUAUAAGAGUUAUAUAUAUAUGCGCGUUCCA
UGGAAGAAGCUCUGGCAGCUUUUUAAGCGUUUAUAUAAGAGUUAUAUAUAUAUGCGCGUUCCA
UGGAAGAAGCUCUGGCAGCUUUUUAAGCGUUUAUAUAAGAGUUAUAUAUAUAUGCGCGUUCCA
UGGAAGAAGCUCUGGCAGCUUUUUAAGCGUUUAUAUAAGAGUUAUAUAUAUAUGCGCGUUCCA

(a)

(b)

Figure 2.3: *(a) shows the base-pair probabilities as predicted by RNAfold [84]. The lower triangle show only the bases included in the minimum free energy structure and the upper triangle contains the full base-pair probabilities where the diameter of a square is proportional to the probability of the corresponding base-pair. (b) shows the 2d plot of the structure annotated with the probabilities of a base-pair. The colors range from blue to red in correspondence to less and high frequent base-pairs.*


(a)

(b)

(c)

(d)

Figure 2.4: *(a) shows a secondary structure with colored components that indicate the relation between the representations. (b) shows the natural tree representation where internal nodes correspond to base-pairs and leaves correspond to unpaired bases. (c) shows the coarse grained tree representation. The red and cyan part are stacking region (S), the green part is a multiloop (M), the yellow part is an internal loop (I), and the blue and magenta parts are hairpin loops (H). A bulge left (L) and a bulge right (R) are internal loops that have only a left and right unpaired region, respectively. Note that single stranded regions at the root level of the tree and in multi-loops are omitted in this tree representation. (d) shows a simplified parse tree for some grammar describing RNA secondary structures. The internal nodes correspond to productions of the grammar and impose a structure on the sequence that resides at the leaves. A virtual root node v is added in (b) and (d) to guaranty a tree structure.*

2.4 RNA Secondary Structure Prediction

The structure of an RNA molecule can be crucial for its function (see Section 1.1). Accordingly, the automatic prediction of RNA structures from sequence information is an important problem. Today, there are two prediction strategies:

- **Thermodynamic approaches**: The conformation of paired and unpaired regions in an RNA structure can be associated with an energy value. Given some energy model, thermodynamic approaches find the energetically most stable structures among all possible secondary structures of an RNA sequence. Such a structure is denoted the *minimum free energy (mfe) structure.*

- **Comparative approaches**: In functional non-coding RNA, the structure of an RNA is conserved during evolution. Since a base-pair can be formed by different combinations of nucleotides, different sequences can have the same or a similar structure. If a family of structural homolog RNA molecules has a sufficient amount of sequence conservation, a multiple sequence alignment can emphasize regions of sequence variation. The regions containing structure-neutral mutations, denoted as *compensatory base changes*, give clues to the structure of an RNA molecule.

In 1978, Nussinov et al. introduced a first folding algorithm requiring a single sequence as input [151]. They determine the structure that maximizes the number of possible base-pairs for an RNA sequence. This problem is also known as the *maximum circular matching problem.* The incorporation of thermodynamics in this model assumes that the energy contribution of each base-pair is independent from adjacent base-pairs in the structure. This assumption is not realistic since the stability of RNA molecules is based on the stacking of base-pairs. Zuker & Stiegler proposed a dynamic programming algorithm that calculates the minimum free energy structure based on a model that considers base-pair stacking and destabilizing loops [247]. Their algorithm uses thermodynamic parameters of Tinoco et al. [201]. The energy model and parameters were refined in [129, 209]. McCaskill introduced a statistically motivated model based on Boltzmann's distribution and thermodynamic parameters, the *partition function* [133]. The most likely structure under this model is the mfe structure. The main contribution of McCaskill is the computation of probabilities for the individual base-pairs. Sakakibara et al. and Eddy & Durbin invented a generalization of hidden markov models, the *stochastic context-free grammars*, and formulated the RNA secondary structure prediction problem in this context [42, 167, 168].

Thermodynamic folding relies on parameters that were measured *in vitro* under fixed conditions which is a simplification of real conditions. The folding *in vivo* takes place in a dynamic, hence, more complex environment. From the inaccuracy of energy parameters (and even the model itself), it is possible that the mfe structure is not the biological correct one. The biological relevant prediction is often a *suboptimal solution* that has an energy close to the mfe structure. Thus, the generation of suboptimal structures is important for the practical impact of prediction algorithms based on thermodynamics [243, 246]. The assumption of equilibrium folding pathways is another common simplification of thermodynamic folding models. Studies of the folding of the *Tetrahymena group I intron* gave insights in the complexity of the folding process [8, 208]. It has been observed that RNA can fold during transcription, the folding process happens on a wide range of time scales, and ions and macromolecules guide the folding. Thus, the kinetics of RNA folding are important to understand the true folding pathway. Models and algorithms for kinetic folding prediction are provided in [49, 70, 138, 232]. A further challenge for mfe folding algorithms are RNA secondary structures that are known to have two conformations depending on some environmental parameters, known as *RNA switches* [126]. Recently, Giegerich et al. provided a structure prediction algorithm based on thermodynamics that compartmentalizes the suboptimal solution space into different *shapes* [59].

The different shapes of an RNA molecule give a compact overview of the structure space and are useful find the biological relevant prediction or to detect different conformational states.

The most popular tools for energy-based RNA secondary structure prediction from single sequences are *mfold* [243, 244] and *RNAfold* [79, 84]. The former implements the mfe algorithm and the latter implement additionally McCaskill's partition function algorithm. Recently, the energy-based prediction of pseudoknotted structures received more attention [35, 158, 161].

Comparative approaches require a set of homologous RNA sequences that have a putative similar structure. The general idea is to exploit the covariance that is expected to occur in aligned stem regions. Until the early 80's the structural inference from homologous RNA sequences had been hand-crafted. Noller & Woese described a procedure to detect compensatory changes in helical elements [147]. An algorithm building upon this strategy was provided by Waterman [224, 225]. Given a multiple sequence alignment, the *mutual information content* and *sequence covariation* are measures that help to automatically identify conserved stem regions [26, 229]. These pure phylogenetic approaches assume that the sequences, in fact, share a common structure, which requires a careful choice of sequences. A combination of phylogenetic information and thermodynamics can further improve the results. A multiple sequence alignment is used to validate predicted structures in [81, 112, 121]. Conversely, Han & Kim resolve ambiguities in the alignment by thermodynamics [73]. As an extension of the minimum free energy approach, *RNAalifold* [83] calculates the best folding using an objective function that combines energy contributions and covariance. Ruan et al.'s *ILM* (iterated loop matching) optimizes a similar objective function [166]. As the name suggests, the structure is iteratively constructed by adding non-conflicting stem regions. *ILM* is capable of returning pseudoknotted RNA structures. Knudsen & Hein predict a common RNA secondary structure by stochastic context-free grammars, implemented in the tool *Pfold* [104].

Sankoff opened a branch of comparative strategies considering the alignment and folding problem simultaneously [172]. The time complexity of Sankoff's algorithm is $O(n^6)$ where n is the length of RNA sequences. This is too high to be practical even for two sequences. Mathews & Turner's *DYNALIGN* restricts the maximum distance of possible base-pairs to bound the parameters that affect time complexity in Sankoff's algorithm [130, 131]. Gorotkin et al.'s *FOLDALIGN* implements a modification of Sankoff's algorithm than does not allow branching structures, which reduces the time complexity. Tabaska & Stormo used a graph theoretic approach, the *maximum weight matching* to infer RNA secondary structures from different sources [189, 190]. They consider a set of base pairing scores that can be derived from a range of sources, such as free energy considerations, mutual information, and experimental data. Hofacker et al. provide a strategy that is based on aligning base-pair probability matrices, predicted by McCaskill's partition function algorithm [80]. Their algorithm is implemented in the tool *pmmulti* in the Vienna RNA package.

According to Gardner & Giegerich, approaches that use phylogenetic information yield significant better predictions than pure thermodynamic approaches [55]. However, the quality of the multiple sequence alignment that should reveal the phylogeny depends on the degree of sequence homology of RNA molecules. The minimum homology that is necessary depends on the particular prediction strategy, i.e. the sources of information that are used to predict structures. Moreover, phylogenetic approaches require a large number of sequences which is a rare situation.

For families of RNA molecules with low sequence conservation, a strategy that was proposed by Shapiro and Konings & Hogeweg more than a decade ago is currently revitalized [105, 180]: First, structures are predicted based on thermodynamics and then a structural alignment, instead of a sequence alignment, is done. Recent progress in structural comparison models and algorithms make this strategy a promising candidate for low sequence homologous, but (putative) structural homologous RNA molecules. In particular, this strategy requires a model for structurally aligning multiple RNA secondary structures. I will provide a structure prediction strategy based on multiple structure alignment in Chapter 5.

2.5 RNA Structure Comparison

The field of RNA structure comparison emerged with the invention of RNA secondary structure prediction algorithms. Since then, the resulting pool of predicted structures, be they right or wrong, were available for analyzing structural properties. The prediction of structural motifs, the inference of a taxonomy based on structural similarity instead of sequence similarity, and the prediction of consensus structures for a set of functionally related RNA molecules are active research topics that involve the comparison of RNA structures. I distinguish the following approaches to compare RNA structures:

- **Base-pair distances**: Base-pair distances are classical mathematical metrics that operate on the base-pair sets of RNA structures.

- **Sequence alignment**: RNA secondary structures are represented as strings that in turn are compared in the sequence alignment model.

- **Edit distances between ordered rooted trees**: Since an RNA secondary structure can be represented as a tree, distances on trees can be applied to compare RNA secondary structures.

- **Arc annotated sequences**: Pure sequence alignment based approaches are extended to incorporate structural constraints that are induced by the structure of RNA. Constrained sequence edit models are generally studied in the context of arc annotated sequences.

- **Graphs**: Graphs can express any sort of RNA structures. Algorithms for the classification of graphs are applied to RNA structure analysis.

Distance and Similarity The result of a comparison of RNA structures can be quantified in two different ways: The first is distance and the second is similarity. Distance measures satisfy the mathematical axioms of a metric (or at least pseudo-metric). A similarity measure assigns a numeric value to some pairs of structures such that the larger the value the more similar the structures are. Distances are non-negative and the distance between two structures is zero iff the structures are equal. In contrast, the similarity of equal structures is an arbitrary positive number. Accordingly, a small distance is equivalent to a large similarity.

In the following sections, I consider distance versions of models for RNA structure comparison. The corresponding similarity versions can be derived easily for distances that are based on optimization problems. For distance problems, optimal means minimal, while for similarity problems optimal means maximal. Throughout this section, $(S_1, P_1), (S_2, P_2), \ldots, (S_n, P_n)$ denote secondary structures.

2.5.1 Base-pair Distances

Base-pair distances are distance measures that are defined on the base-pair sets of RNA structures. An analysis of some properties of base-pair distances and their comparison with the tree edit distance is provided in [142].

Symmetric Set Difference

One of the simplest measures is defined by the *symmetric set difference*, that is:

$$\delta_{\mathrm{SD}}(P_1, P_2) = P_1 \setminus P_2 \cup P_2 \setminus P_1 \tag{2.3}$$

Clearly, this simple measure is sensitive to the exact position of base-pairs and is therefore not suitable to compare structures of different length. Also if the structures have the same length, the measure is sensitive for shifted structures. Consider the following structures:

$$
\begin{aligned}
P_1 &= \texttt{............(((.....))).}\\
P_2 &= \texttt{...........(((.....)))..}
\end{aligned}
$$

Intuitively, these structures should obtain a distance close to zero, but $\delta_{\mathrm{SD}}(P_1, P_2) = 6$ since there is no common base-pair. This discrepancy gets the larger the larger the shifted structures are. Still, for suboptimal structures of the same sequence, δ_{SD} can be a useful *ad hoc* distance.

Hausdorff Distance

A more flexible metric is the *Hausdorff distance* which was applied by Zuker to filter out similar suboptimal foldings in the original *mfold* program [243]. The Hausdorff distance measures the distance between non empty point sets of some metric space. For the problem of RNA structure comparison, these are the sets of base-pairs. Intuitively, the Hausdorff distance between structures P_1 and P_2 is the maximum of the distances between all nearest base-pairs connecting P_1 and P_2. Formally, the distance between two base-pairs $(i, j) \in P_1$ and $(i', j') \in P_2$ is defined as $\delta((i, j), (i', j')) = max\{|i-i'|, |j-j'|\}$. The distance of a base-pair to a set of base-pairs is defined as $\delta((i, j), P) = \inf_{(i',j')\in P} \delta((i, j), (i', j'))$. Then the *Hausdorff distance between P_1 and P_2* is defined as

$$
\begin{aligned}
\delta_{\mathrm{H}}(P_1, P_2) &= \max(\delta_{\mathrm{asym}}(P_1, P_2), \delta_{\mathrm{asym}}(P_2, P_1)) \text{ where} \qquad (2.4)\\
\delta_{\mathrm{asym}}(P_1, P_2) &= \sup_{(i,j)\in P_1} \delta((i, j), P_2)
\end{aligned}
$$

Although this distance behaves reasonable for structure shifts, the distance between structures that differ only in one base-pair depends on the position of this base-pair. Consider the following structures:

$$
\begin{aligned}
P_1 &= \texttt{...........(((.....)))..}\\
P_2 &= \texttt{(...)......(((.....)))..}\\
P_3 &= \texttt{....(...)..(((.....)))..}
\end{aligned}
$$

P_2 and P_3 are both one base-pair apart from P_1, but their Hausdorff distance is different, i.e. $\delta_{\mathrm{H}}(P_1, P_2) = 11$ and $\delta_{\mathrm{H}}(P_1, P_3) = 7$. Thus, isolated base-pairs can lead to high distance values depending on the distance to the next base-pair. Aware of this problem, Zuker et. al defined a variant of δ_{H} that ignores up to d bases to obtain a distance d [246]. This variant is a pseudo-metric, since the triangle inequality is not satisfied as exemplified in [142].

Mountain Metric

Another application of a classical mathematical metric to RNA structures is the *mountain metric* which is based on the l_p-norm of the difference of two mountain functions h_{P_1} and h_{P_2} (see Equation (2.2)) of RNA secondary structures of length n [142]:

$$\delta_{\mathrm{M}}^{p}(P_1, P_2) = \|h_{P_1} - h_{P_2}\|_p := \sqrt[p]{\sum_{i=1}^{n} |h_{P_1}(i) - h_{P_2}(i)|^p} \tag{2.5}$$

For $p = 2$ this is the *root mean square (RMS) distance* between two functions which is, followed by $p = 1$, the most frequent choice. This metric is more flexible for shifted structures and isolated base-pairs and it can be computed in linear time. A property of this distance that one must be aware of is that the extension of stem regions does not have uniform costs. See the following example:

$$\begin{aligned} P_1 &= \texttt{..(((.....)))..} \\ P_2 &= \texttt{.((((.....)))).} \\ P_3 &= \texttt{..((((...))))..} \end{aligned}$$

P_1 differs from P_2 and P_3 in just one base-pair but their mountain distances (for simplicity I use $p = 1$) do not reflect that. In particular, $\delta_{\mathrm{M}}^{1}(P_1, P_2) = 13$ and $\delta_{\mathrm{M}}^{1}(P_1, P_3) = 5$. See Figure 2.5 for an illustration. A variant of the mountain distance that re-scales mountain functions for structures of different length is proposed and applied in [44].

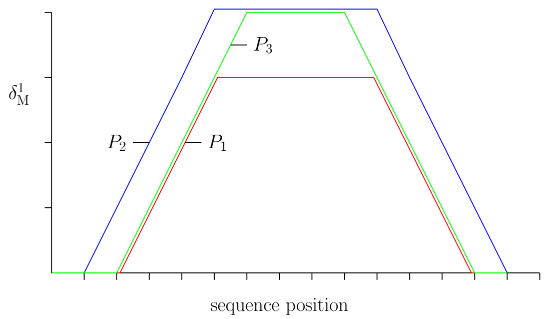


Figure 2.5: *The difference between P_2 and P_1 is larger than the difference between P_3 and P_1, though both differ in exactly one base-pair.*

2.5.2 Sequence Alignment

Shapiro and Konings & Hogeweg simultaneously proposed the idea to compare RNA secondary structures by well established sequence alignment algorithms [105, 180]. While Konings & Hogeweg focused on pairwise alignments, Shapiro considered multiple sequence alignments. In both approaches, the key idea is to use a string representation of RNA secondary structures, in flavor of the Vienna strings[1], which are the data structures that are further analyzed.

Konings & Hogeweg's Encoding

Following Konings & Hogeweg, "A full linear representation is obtained by transforming the mountain structure into a linear array of symbols representing the direction of base-pairing at each of the single positions: upstream pairing (>), downstream pairing (<) or single strandedness (+) ... Extra information in terms of secondary structure can be included in the linear representation by distinct coding of hairpin loops (^) and other types of single stranded positions (+)". In this representation, the secondary structure in Figure 2.2 is written as:

```
+>>>+>>>>>>+++++<<<<<<^^^^>>>>+>>>>>>^^^^^^^<<<<<<+<<<<++<<<+
```

[1] The Vienna format was established later, building upon the results of Shapiro and Hogeweg & Konings.

A potential disadvantage of this representation for a topological classification is that basic secondary structure elements may be broken up in an alignment, i.e. matching of individual parts of one helix to parts of two different helices, not considering interruptions by internal loops and bulges.

Shapiro's Encoding

Shapiro introduced a different string representation that circumvents this problem. The coarse grained tree representation of an RNA structure is transformed to a string by a left-to-right preorder traversal of the tree, putting subtrees into brackets. The components are encoded as single letters. In this representation the structure in Figure 2.2 is:

```
(S(M((H)(S(I(H)))))
```

To simplify the notation brackets are removed for non-branching subtrees:

```
(S(M(H (S I H))))
```

For a topological classification, this coarse grained representation is suitable. However, if the aim is an improved sequence alignment that incorporates structural constraints, it should be possible to match individual parts of one helix with two different parts of another helix. For instance, there could exist a larger helix that was broken during evolution resulting in two smaller helices that are separated by a bulge.

Beside these effects, both methods suffer from the same inherent problem: A pair of brackets is not treated as a unit by a sequence alignment and thus the tree nature of a secondary structure is not treated appropriately. Consider the following structures:

$$P_1 = \texttt{(((..(((....))))))}$$
$$P_2 = \texttt{(((......)))}$$

The following alignment is among the optimal alignments given a scoring scheme that favors matches in contrast to mismatches, insertions and deletions.

```
(((..(((....))))))
(((..---....)))---
```

The opening brackets ‘(’ are not aligned to its corresponding closing brackets ‘)’ and in terms of structure this alignment is not meaningful. Shapiro was aware of such problems but appropriate, efficient algorithms for comparing RNA secondary structures as trees were just about to emerge.

2.5.3 Edit Distances between Rooted Ordered Trees

From the tree nature of RNA secondary structures, every distance measure on trees can be applied to RNA secondary structures. Inspired by the sequence edit distance [113, 171, 213], different edit models for trees have been invented [95, 118, 177, 191, 198] which result in various algorithms. Beside the fact that tree editing is a challenging theoretical problem dealing with a fundamental data structure, this field was (and is still) driven by the need for such algorithms in a broad spectrum of applications. This includes the comparison of RNA secondary structures [25, 110, 111, 178], the analysis of structured documents and text databases [18, 96, 127, 144, 159], script recognition [22, 118], fingerprint recognition [139], image analysis [165, 169], the analysis of parse trees [97, 235], the comparison of assembly rules [48], and the identification of common structural fragments among chemical structures [192]. The semantic of tree edit distances in the scope of RNA structure comparison depends on the choice of the tree representation and the edit model.

A review of tree edit models that are particularly interesting for document trees (but also for RNA secondary structures) was given in [7]. The authors provide implementations of tree edit algorithms in the programming language *Turing* [90]. A more recent survey on tree editing problems, including unrooted, unordered variants, and different notions of tree editing, was provided in [10, 11, 241]. The relation between tree-edit distances was studied in [216] resulting in a hierarchy of edit-models.

In the world of sequences, the terms *edit distance* and *alignment distance* are used synonymously. For each optimal sequence of edit operations, an alignment achieving the same score can be constructed and vice versa. However, on a conceptional level the models are different. While the edit distance is an operational model of editing one sequence into another, an alignment is a declarative model, a data structure rather than a process. In the world of trees, these models turned out to be dual: The tree edit model constructs a largest common subforest, while the tree alignment distance constructs a smallest common supertree. Moreover, the higher complexity of trees (in comparison to sequences) leads to a multitude of problems that vary in the constraints that are imposed by the chosen model. The models that are interesting for the comparison of RNA structures are introduced in the following paragraphs, beginning with the most general model which is successively restricted. Throughout this chapter, T, T_1, T_2 are trees unless stated differently.

Tree Edit Distance

In *the tree-to-tree correction problem* [191], Tai introduced the generalization of *the string-to-string correction problem* [213] which is also known as the *edit distance problem for strings*. I refer to Tai's model as the *tree edit model*[2], following the mainstream of literature.

Edit Operations The edit operations *relabel*, *delete* and *insert* generalize from strings to trees (and forests) as follows:

- *relabel*: The label of a node v in T is changed. If a label is relabeled by itself, this is denoted a *match*.
- *delete*: Deleting node v in T means that the children of node v become the children of the parent node of v. Moreover, if v has any siblings, the deletion preserves the preorder relation of these node. Note, if v is the root node, the result is the forest consisting of the children nodes of v.

[2] The same model was also, independently, proposed by Lu [118]. However, Lu considered an algorithm for a special case of the general tree edit distance.

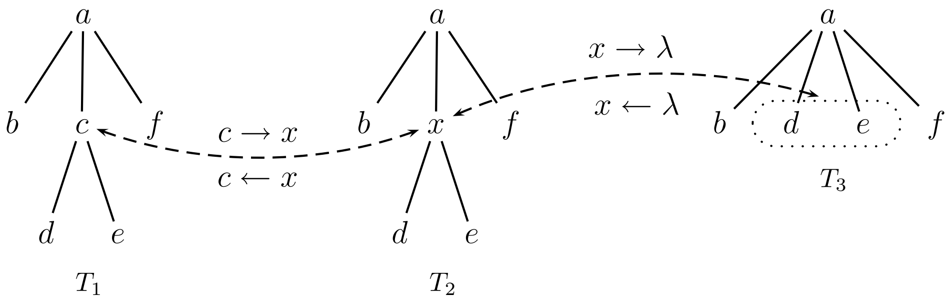


Figure 2.6: *To simplify the illustration, a node and its label are identical. T_1 is transformed into T_2, by relabeling c with x, which in turn is transformed into T_3 by deleting x. Note that the edit operations can be applied in both directions. T_2 results from T_3 by inserting x as a child of node a whereas the nodes d and e become the children of x.*

- *insert*: This operation is complementary to *delete*. Inserting a new node v into T results in a new tree T' such that the deletion of v in T' results in T. Intuitively, a node v is inserted as a child of v' making v the parent of a consecutive subsequences of children of v'.

According to the sequence edit model, I represent edit operations by $\alpha \rightarrow \beta$ where $(\alpha, \beta) \in \Sigma_\lambda^2$. $\alpha \rightarrow \lambda$ and $\lambda \rightarrow \beta$ denote the functions *delete* and *insert* of a and b, respectively. Otherwise, $a \rightarrow b$ is the *relabel* function, relabeling a with b. An illustration of the tree edit operations is given in Figure 2.6. Note, the node that is affected by an edit operations is defined by the edit operation together with the tree to be edited and the resulting tree.

Let E be a sequence $e_1, e_2, \ldots, e_n$ of edit operations, for short *edit-sequence*. Following Tai, E transforms T into T' if there is a sequence of trees $T_0, T_1, \ldots, T_n$ such that $T = T_0, T' = T_n$ and T_i results from the application of e_i to T_{i-1} for $i \in [1, n]$. Let δ be a metric defined on edit operations. The cost of an edit-sequence E is the sum of the costs of its edit operations, that is: $\delta(E) = \sum_{i=1}^{n} \delta(e_i)$ which is also a metric [240]. The *edit distance* δ_{TE} between trees T_1 and T_2 is the minimum cost that is necessary to transform T_1 into T_2:

$$\delta_{\mathrm{TE}}(T_1, T_2) = \min\{\delta(E) \mid E \text{ is an edit sequence transforming } T_1 \text{ into } T_2\}. \tag{2.6}$$

Edit sequences are an intuitive, operational concept that accounts for the differences between trees. However, the infinite number of edit sequences that can transform one tree into another make theoretical observations intricate. Again inspired by the sequence edit model, Tai extended the concept of *traces*, known from the sequence edit model [213], to trees, commonly referred to as *mappings*.

Mappings A mapping establishes a one-to-one correspondence of nodes in T_1 and T_2 which preserves the sibling and ancestor relation of nodes. Formally, a *mapping* between trees T_1 and T_2 is defined by a triple (M, T_1, T_2) where $M \subseteq V(T_1) \times V(T_2)$ such that for all $(v_1, w_1), (v_2, w_2) \in M$ the following holds:

$$v_1 = v_2 \text{ iff } w_1 = w_2 \qquad \text{(one-to-one correspondence)}$$

$$v_1 \text{ is ancestor of } v_2 \text{ iff } w_1 \text{ is ancestor of } w_2 \qquad \text{(ancestor preservation)}$$

$$pre_{T_1}(v_1) < pre_{T_1}(v_2) \text{ iff } pre_{T_2}(w_1) < pre_{T_2}(w_2) \qquad \text{(sibling preservation)}$$

Let $V(T_1) \setminus M$ and $V(T_2) \setminus M$ be the nodes in T_1 and T_2 that are not mapped by M, respectively. The cost of a mapping is given by:

$$\delta(M) = \sum_{(v,w)\in M} v \to w + \sum_{v \in V(T_1)\setminus M} v \to \lambda + \sum_{w \in V(T_2)\setminus M} \lambda \to w \tag{2.7}$$

The following lemma shows that mapping are equivalent to edit-sequences.

Lemma 2.1. *Given an edit-sequence E transforming T_1 into T_2, there exists a mapping from T_1 to T_2 such that $\delta_{TE}(M) \leq \delta_{TE}(E)$. Conversely, for any mapping M, there exists an edit-sequence such that $\delta_{TE}(E) = \delta_{TE}(M)$.*

Proof. See Proof of Lemma 2 in [240]. □

Hence, the edit distance between trees can be defined likewise by

$$\delta_{\mathrm{TE}}(T_1, T_2) = \min\{\delta(M) \mid M \text{ is a mapping from } T_1 \text{ to } T_2\}. \tag{2.8}$$

Isomorphic Subforests A third definition of the edit distance between trees is more related to graph theory. Forests F_1 and F_2 are *isomorphic*, denoted by $F_1 \cong F_2$ if they can be transformed into each other simply by applying the *relabel*-function. For isomorphic forests, there exists a corresponding mapping M_i including all nodes in F_1 and F_2. Such a mapping M_i is denoted an *isomorphism.* For some $D \subseteq V(T)$, $T \setminus D$ denotes the *forest* that results from applying the *delete*-function to all nodes in D to T. This definition, allowing isomorphic subforests instead of isomorphic subtrees, is important since a valid mapping between trees can correspond to an isomorphic subforest. The edit distance between T_1 and T_2 can then be defined as

$$\begin{aligned}\delta_{\mathrm{TE}}(T_1, T_2) = \min\{\delta_{\mathrm{TE}}(M_i) + \sum_{v \in D_1} v \rightarrow \lambda + \sum_{w \in D_2} \lambda \rightarrow w \mid \\ D_1 \in V(T_1), D_2 \in V(T_2) \text{ such that } T_1 \setminus D_1 \cong T_2 \setminus D_2\}. \end{aligned} \tag{2.9}$$

It is obvious that this definition is equivalent to the definition of a mapping (2.8) and, consequently, to the edit sequence based definition. Figure 2.7 shows an example of a mapping and the correspondence to isomorphic subforests.

Algorithms Algorithms that calculate the tree edit distance generally build upon the mapping concept since the number of mappings for given trees is finite. The first proposed algorithm is due to Tai and requires $O(|T_1| \cdot |T_2| \cdot \mathit{leaves}(T_1)^2 \cdot \mathit{leaves}(T_2)^2)$ time and space. It follows the strategy of extending mappings from the root of a tree to its leaves. A faster and much simpler algorithm is due to Zhang & Sasha (Zhang-Shasha Algorithm) and improves the time complexity to $O(|T_1| \cdot |T_2| \cdot \min\{\mathit{leaves}(T_1), \mathit{depth}(T_1)\} \cdot$

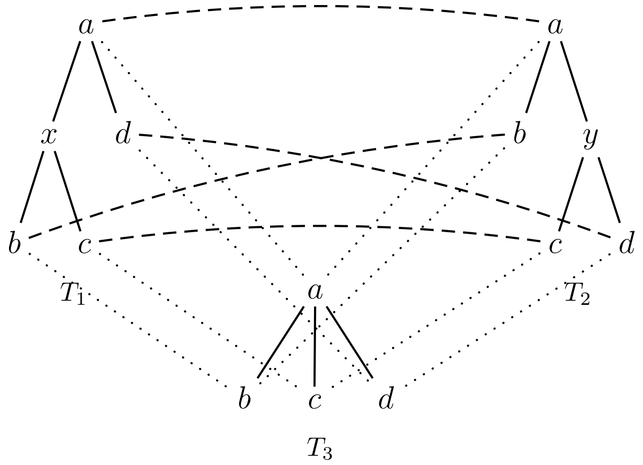


Figure 2.7: *The dashed lines indicate the mapping* $M = \{(a,a),(b,b),(c,c),(d,d)\}$ *of* T_1 *and* T_2. T_3 *shows the maximum isomorphic subforest (here a tree) that is obtained by deleting node* x *in* T_1 *and node* y *in* T_2*. The edit sequence* $x \to \lambda, \lambda \to y$ *together with the sequence of trees* T_1, T_3, T_2 *determines the corresponding edit process.*

$\min\{leaves(T_2), depth(T_2)\})$ and the space complexity to $O(|T_1| \cdot |T_2|)$ [240]. In the worst case, which is a tree that grows linear in the number of leaves and its depth, the time complexity is in $O(|T_1|^2 \cdot |T_2|^2)$. Special algorithms for the tree edit distance under a unit cost scheme are studied in [181]. The parallelization of tree edit algorithms is considered in [237, 239]. The average runtime of the Zhang-Shasha Algorithm for RNA secondary structure trees turned out to be $O(|T_1|^{\frac{3}{2}} \cdot |T_2|^{\frac{3}{2}})$ which essentially means that it is cubic [39]. Klein improved the worst case runtime of the tree edit algorithm to $O(|T_1|^2 \cdot |T_2| \cdot \log|T_2|)$ by applying a divide and conquer strategy (Klein's Algorithm) [102]. An analysis of the Zhang-Shasha Algorithm and Klein's Algorithm in a general framework of *cover strategies* is given by Dulucq & Touzet [40]. Moreover, they present an improvement of Klein's strategy which can result in a better practical runtime. A different strategy is followed by Chen, the tree edit problem is reduced to a matrix multiplication problem and is solved by using results in this field [21]. This algorithm runs in $O(|T_1| \cdot |T_2| + \min\{leaves(T_1)^2 \cdot |T_2| + leaves(T_1)^{2.5} \cdot |T_2|, leaves(T_2)^2 \cdot |T_1| +$

$\mathit{leaves}(T_2)^{2.5} \cdot |T_1|\})$ and improves the time complexity for certain kind of trees in comparison to Klein's algorithm, e.g. if one of T_1 and T_2 is thin and deep.

Variants Touzet gave a definition of gaps in a tree [207]. The idea is to consider contiguous gaps as a single large gap where the term contiguous is equivalent to our definition of a tree pattern. They study convex scoring functions for gaps, that is: $\mathit{gapscore}(T_1 \circ T_2) \leq \mathit{gapscore}(T_1) + \mathit{gapscore}(T_2)$ where T_1 and T_2 are tree patterns and $T_1 \circ T_2$ means that T_2 is attached to a leaf node of T_1. They proved that the calculation of the tree edit distance with gaps for convex gap scores is a NP-hard problem.

Tree Alignment Distance

The tree alignment distance was introduced by Jiang et al. [95]. My central notion is the following generic view of an alignment: An *alignment* of two structures with labels from some alphabet Σ is the same type of structure with labels from the alignment alphabet Σ^2_λ. Labels of the form $(\alpha, \beta), (\alpha, \lambda), (\lambda, \beta)$ where $\alpha, \beta \in \Sigma$ denote the edit operations *relabel*, *delete*, and *insert*, respectively. Applying this general concept to trees, a tree alignment A is an element of $\mathcal{T}(\Sigma^2_\lambda)$. Its component-wise projections $A|_1$ and $A|_2$ are elements of $\mathcal{T}(\Sigma \cup \{\lambda\})$. For some $T \in \mathcal{T}(\Sigma \cup \{\lambda\})$, $\pi(T) \in \mathcal{F}(\Sigma)$ is the forest that results from the deletion of all nodes v with $\mathit{label}(v) = \lambda$. Formally[3]:

$$\pi(T) = T \setminus D \text{ where } D = \{v \mid \mathit{label}(v) = \lambda\} \tag{2.10}$$

The following equation formally defines the notion of alignment of trees.

$$A \in \mathcal{T}(\Sigma^2_\lambda) \textit{ is an alignment of trees } T_1, T_2 \in \mathcal{T}(\Sigma) \text{ iff}$$
$$T_1 = \pi(A|_1) \text{ and } T_2 = \pi(A|_2). \tag{2.11}$$

[3] See the definition of $T \setminus D$ on Page 34.

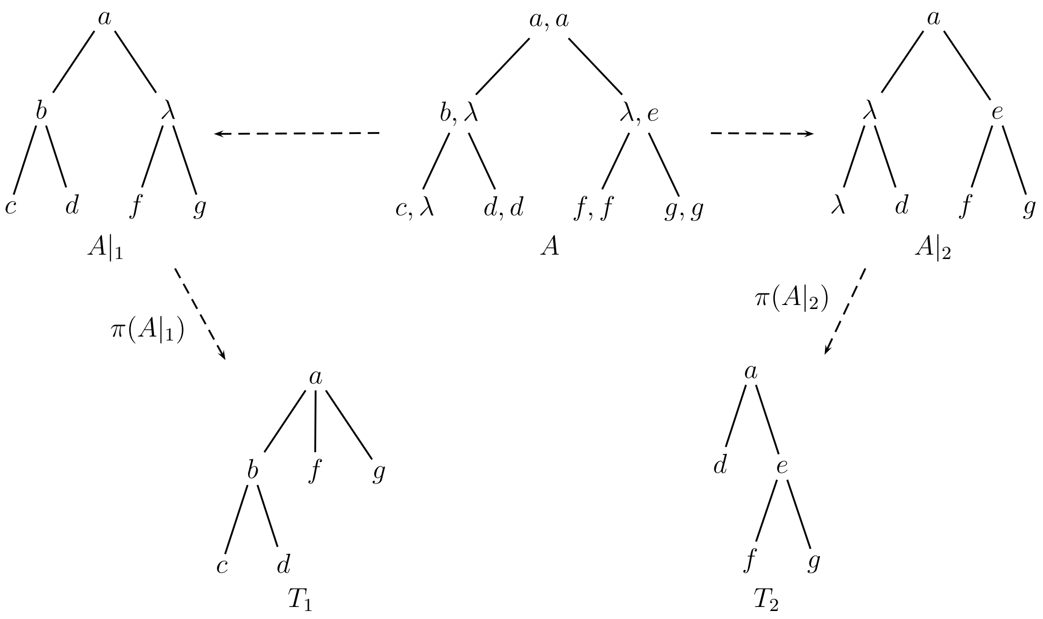


Figure 2.8: *A is an alignment of T_1 and T_2.*

Note that this definition forbids elements of $\mathcal{T}(\Sigma^2_\lambda)$ where the deletion of a root node results in a forest (A forest alignment model will be introduced in Section 3.2). Figure 2.8 shows an example of a pairwise tree alignment. The cost δ of an alignment A is the sum of the costs of its node labels, that is:

$$\delta(A) = \sum_{v \in V(A)} \delta(label(v)). \qquad (2.12)$$

The alignment distance between T_1 and T_2 is the minimum cost that an alignment of T_1 and T_2 can achieve. An alignment of T_1 and T_2 is *optimal* if it achieves this score. Formally, the *alignment distance* δ_{TA} between trees T_1 and T_2 is defined as:

$$\delta_{\text{TA}}(T_1, T_2) = \min\{\delta(A) \mid \text{A is an alignment of } T_1 \text{ and } T_2\} \qquad (2.13)$$

For each alignment it is possible to construct a corresponding edit sequence and a mapping. The converse does not hold in general: Consider the mapping in Figure 2.7. In this mapping, nodes labeled with "c" are mapped to each other. Thus, in a possible alignment there must exist a node labeled with "c, c". Then, this node must be the son of the nodes labeled with "x, λ" and "λ, y". This is in contrast to the definition of a tree since a node can have at most one parent node in a tree. From this observation, it is clear that tree alignments form a subset of tree edit distance mappings. For trees T_1 and T_2 holds $\delta_{\mathrm{TE}}(T_1, T_2) \leq \delta_{\mathrm{TA}}(T_1, T_2)$.

Since the edit sequence definition is equivalent to the mapping definition, it follows that not each edit sequence has a corresponding alignment. Jiang et al. claimed that an "alignment of trees actually corresponds to a restricted tree edit in which all the insertions precede all the deletions" [95]. This is intuitive, but a formal proof is missing.

I now demonstrate that δ_{TA} does not satisfy the triangle inequality of the metric axioms: An arbitrary edit sequence can be divided into two edit sequences where the one includes all insert- and the other all delete- and relabel-operations. Assuming Jiang et al.'s claimed property of alignment compatible edit sequences (see above), the divided edit sequences are compatible with an alignment. From this and the fact that the tree edit distance can be less than the tree alignment distance follows that it does not satisfy the triangle inequality. Hence, the tree alignment distance is not a metric. See Figure 2.9 for an example.

I am not aware of a constrained mapping definition that corresponds to alignments, in literature.

Isomorphic Supertree A graph theoretical definition of the tree alignment distance is based on tree isomorphisms. In this context, the minimum possible distance between isomorphic trees that result from the insertion of "λ" labeled nodes in the original trees is sought. The forests that are considered by this procedure are *isomorphic supertrees*. Nodes that are labeled with "λ, λ" should naturally score 0. Clearly, an overlay of such isomorphic superforests and the deletion of possible "λ, λ" labeled nodes produces an alignment and, hence, the models define the same distance.

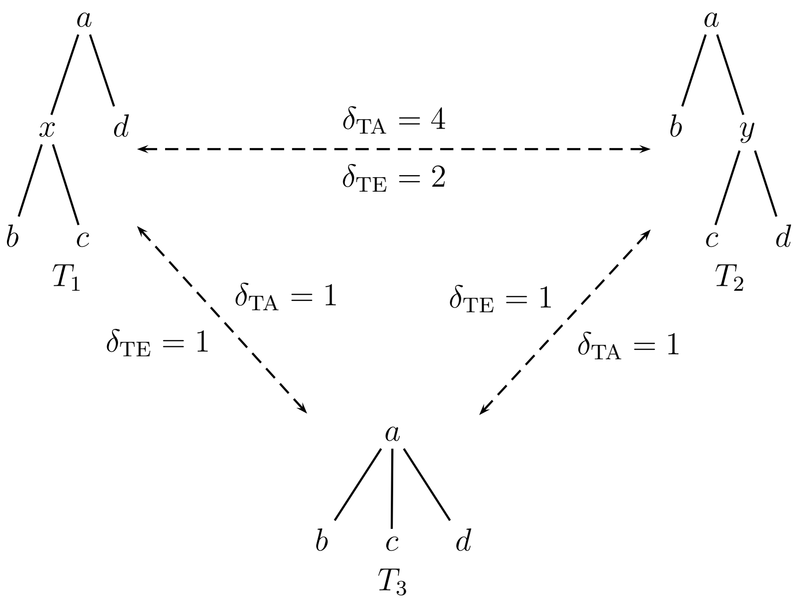


Figure 2.9: *Consider the unit cost function, the triangle inequality of the tree alignment distance is not satisfied since $\delta_{TA}(T_1, T_2) \nleq \delta_{TA}(T_1, T_3) + \delta_{TA}(T_3, T_1)$. In the tree edit model the triangle inequality is satisfied.*

Algorithms Together with the definition of the tree alignment distance, Jiang et al. proposed an algorithm that computes this distance in $O(|T_1| \cdot |T_2| \cdot (degree(T_1) + degree(T_2))^2)$ time which is still the asymptotical best algorithm [95]. For a fixed number d of possible deletions and insertions, Jansson & Lingas presented an algorithm that calculates the tree alignment distance[4] in $O(n^2 \cdot \log n \cdot k^3 \cdot d^2)$ where $n = \max\{|T_1|, |T_2|\}$ and $k = \max\{degree(T_1), degree(T_2)\}$ [92].

Variants Wang & Zhao make three interesting contributions considering the tree alignment distance for RNA structure comparison [221]:

1. They provide a model for the tree alignment distance including gaps where the notion of gaps in a tree corresponds to tree patterns as done in [207]. However, Wang & Zhao consider a simpler gap score function where the score of a gap is a constant function. They derive

[4] Precisely, the similarity version.

an algorithm from Jiang et al.'s algorithm that computes the alignment distance, involving gap scores, in the same time complexity.

2. They present a modified version of Jiang's algorithm that improves the space complexity to $O(\mathit{degree}(T_1)\cdot\log|T_1|\cdot|T_2|\cdot(\mathit{degree}(T_1)+\mathit{degree}(T_2)))$ while having the same time complexity as the Jiang algorithm. However, an optimal alignment can not be obtained by a straightforward backtracking procedure. As space is crucial in their application they use a naive algorithm that raises the time complexity to $O(|T_1|^2 \cdot |T_2| \cdot (\mathit{degree}(T_1) \cdot \mathit{degree}(T_2))^2)$ while achieving their improved space complexity.

3. They consider the problem of parametric tree alignment which was studied earlier for sequences [71] and gives clues to the parameter space of tree alignments. In particular, the scoring of edit operations is often not deducible from the problem and therefore somewhat arbitrary. Parametric alignment partitions the parameter space into regions such that in each region any alignment, that is optimal for some choice of parameters inside the region, is optimal throughout that entire region and nowhere else. A software to visualize and explore the parameter space is also provided.

Isolated Subtree Distance

The isolated subtree distance was first proposed in [198][5] and is also referred to as the *structure respecting edit distance* or *structure preserving mapping distance*. Intuitively, it restricts mappings such that two separate subtrees in T_1 are mapped to two separate subtrees in T_2. Alternatively formulated, trees can only be mapped to trees and not to forests.

[5] In [198], Tanaka & Tanaka refer to an earlier publication that introduce this distance [197]. As it is written in Japanese I was not able to validate this. Further early contributions in the field of tree editing, again in Japanese, are given in [1, 193–196].

Mappings A mapping M between trees T_1 and T_2 is an *isolated subtree mapping* if for all $(v_1, w_1), (v_2, w_2), (v_3, w_3) \in M$ holds:

$$lca(v_1, v_2) = lca(v_1, v_3) \text{ iff } lca(w_1, w_2) = lca(w_1, w_3) \qquad \text{(isolated subtree condition)}$$

The *isolated subtree distance* δ_{TI} between T_1 and T_2 is the minimum cost that an isomorphic subtree mapping between them can achieve. Formally,

$$\delta_{\mathrm{TI}}(T_1, T_2) = \min\{\delta(M) \mid M \text{ is an isolated subtree mapping between } T_1 \text{ and } T_2\}. \quad (2.14)$$

Figure 2.10 shows an example of a mapping that is not an isolated subtree mapping, but corresponds to an alignment. The metric properties of the isolated subtree distance are proven in [236].

Algorithms Tanaka & Tanaka proposed an algorithm that computes the isolated subtree distance in $O(|T_1| \cdot |T_2| \cdot \min\{leaves(T_1), leaves(T_2)\})$ time and $O(|T_1| \cdot |T_2|)$ space [198]. Zhang improved the worst case complexity to $O(|T_1| \cdot |T_2|)$ time and space [236]. Later, Richter presented an algorithm that computes the isolated subtree distance in $O(|T_1| \cdot |T_2| \cdot degree(T_1) \cdot degree(T_2))$ time and $O(|T_1| \cdot depth(T_2) \cdot degree(T_2))$ space. For balanced trees of bounded degree k, i.e. each internal node has k children, this algorithm consumes less space than Zhang's Algorithm.

Top-Down Distance

Although I introduce the top-down distance at the end of this survey, its introduction by Selkow opened the discipline of tree edit distances in 1977 [177]. He considered a tree edit distance model where insertions and deletions are restricted to the leaves of a tree: Only leaves may be deleted, and a node may be inserted only as a son of a leaf.

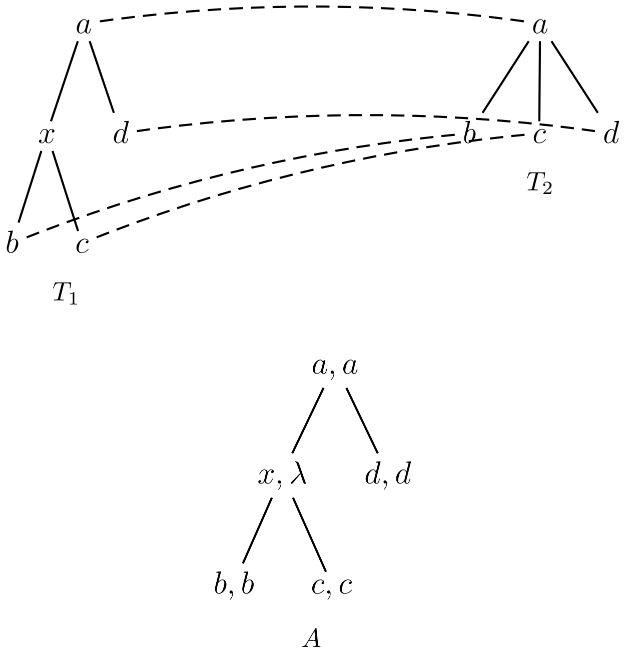


Figure 2.10: *The mapping between T_1 and T_2 is not an isolated subtree mapping, since it violates the isolated subtree condition. In particular, for T_1 holds lca$(b, c) \neq$ lca(b, d) but for T_2 holds lca$(b, c) =$ lca(b, d). Even this mapping is not a valid isolated subtree mapping, there exists a corresponding alignment A.*

Mappings In terms of mappings, this has the consequence that whenever w.l.o.g a node v in T_1 is mapped to some node in T_2, all ancestor nodes of v must be included in the mapping. Given some mapping M between T_1 and T_2, let $M|_1$ and $M|_2$ be the nodes in T_1 and T_2 that are touched by M, respectively. Let $ancs_T(v)$ denote the set of all ancestor nodes of v. Formally, a mapping M between trees T_1 and T_2 is a *top-down mapping* if the following holds:

$$(v, w) \in M \Rightarrow ancs_{T_1}(v) \subseteq M|_1 \text{ and } ancs_{T_2}(w) \subseteq M|_2 \tag{2.15}$$

The *top-down distance* δ_{TD} between T_1 and T_2 is the minimum cost that an top-down mapping between them can achieve:

$$\delta_{\mathrm{TD}}(T_1, T_2) = \min\{\delta(M) \mid M \text{ is a top-down mapping between } T_1 \text{ and } T_2\} \tag{2.16}$$

Recently, Valiente proposed a “dual” model, a *bottom-up distance* between Trees, where deletions and insertions must begin at the root level [210].

Algorithms Selkows algorithm computes the top-down distance in $O(|T_1| \cdot |T_2|)$ time and space [170, 177]. The algorithm was implemented and applied to the problem of identifying syntactic differences in [235].

2.5.4 Related Problems

Similar Consensus Problems

The similar consensus problem is the problem of finding a largest approximately common substructure in trees. For strings, a substructure is a subword. For graphs, a substructure can be defined as a connected subgraph which for trees results in my definition of a tree pattern. Let d be an integer, the *similar consensus problem* is to find pattern trees T_1' of T_1 and T_2' of T_2 such that the distance between T_1' and T_2' is within distance d and there does not exists any other substructure T_1'' of T_1 and T_2'' of T_2 that satisfy the distance constraint and $|T_1'| + |T_2'| \leq |T_1''| + |T_2''|$. The similar consensus problem was studied for the different distances that were presented in this section:

distance	time complexity	studied in
tree edit distance	$O(d^2 \cdot \|T_1\| \cdot \|T_2\| \cdot C(T_1) \cdot C(T_2))$, where $C(T_i) = \min\{\mathit{leaves}(T_i), \mathit{depth}(T_i)\}$	[214]
tree alignment distance	$O(\|T_1\| \cdot \|T_2\| \cdot (\mathit{degree}(T_1) + \mathit{degree}(T_2))^2)$	[215]
isolated subtree distance	$O(d^2 \cdot \|T_1\| \cdot \|T_2\|)$	[216]
top-down distance	$O(d^2 \cdot \|T_1\| \cdot \|T_2\|)$	[217]

Tree Inclusion Problems

The tree inclusion problem is a variant of the general tree edit distance. In terms of a maximum isomorphic subtree, a tree pattern T_p is included in a target tree T if T_p can be obtained from T by node deletions. This corresponds to an edit model that only supports the functions *relabel* and *insert* where T_p is the first and T the second tree. Kilpäinen & Mannila presented an algorithm that solves this problem in $O(|T_p| \cdot |T|)$ [98]. Improvements and variations of their algorithm are proposed in [3, 20, 160]. The classic problem of *tree pattern matching* is a restricted version of the tree inclusion problem. The deletion of nodes in the target tree is only allowed for leaf nodes in T (and the trees that result from such deletions), which is equivalent to subtree removals in T. This corresponds to the tree inclusion problem in the domain of Selkow's top-down distance. Among others, substantial contributions are reported in [28, 38, 86, 106, 119, 125, 157].

Zhang et al. considered the approximate tree matching in the presence of variable length don't cares (VLDC) [219]. The query tree can contain wildcards that may match multiple nodes. For example, symbol "|" substitutes for a part of a path from the root to a leaf in the target tree. Symbol "^" matches a path and all subtrees emanating from the nodes on that path. Building upon that wildcards, the authors introduced a querying language for inexact matching of trees.

2.5.5 Arc Annotated Sequences

The pure sequence based approaches to compare RNA secondary structures are known to have the problem of violating the tree structure (see Section 2.5.2). On the other hand, tree edit based approaches are so far limited to compare RNA secondary structures. Moreover, in the coarse grained tree representation the meaning of tree edit operations in the process of editing RNA structures is difficult to motivate biologically. In the natural tree representation, the tree edit model cannot account adequately for a deletion of a base-pair bond. This gave rise to the idea of incorporating structural constraints into sequence alignment strategies.

The first structural refined sequence alignment algorithm was proposed by Sankoff [172], although for the more sophisticated problem of folding and aligning simultaneously. Bafna et. al. introduced the concept of *RNA strings* which include both, the primary sequence and the secondary structure information [4]. Beside matching problems on RNA strings, they introduced an alignment model for RNA strings. Evans generally studied annotation schemes that add auxiliary information to a sequence. These can be taken into account when the sequences are analyzed [45]. Evans introduced the general notion of *arc-annotated sequences*. An *arc* is a link joining two different symbols of a sequence and can be used to represent a binary relation between them. The definition of an *arc-annotated sequence* complies to the definition of a tertiary structure[6] (see Section 2.2). As a natural extension of the *longest common subsequence problem*, Evans introduced the *longest arc-preserving common subsequence problem* [45]. This problem is not only studied extensively due to its potential application for RNA structure comparison, but also because it has a compact definition, is easy to understand and turned out to be NP-hard even for RNA secondary structures [114]. Zhang et al. introduced a further edit model for RNA structures including tertiary interactions [242]. For RNA secondary structures, their model corresponds to the tree edit model in conjunction with the natural tree representation. Finally, Jiang et al. suggested a set of edit operations for RNA structures that are biological motivated and form a superset of edit operations of the formerly mentioned models [94]. I introduce this general edit model for RNA structures first and use its terminology to give a uniform description of the other models.

[6] A general arc-annotated structure additionally allows a connection of one to many characters. I neglect this case since complex interactions like base-triplets are beyond the scope of this thesis.

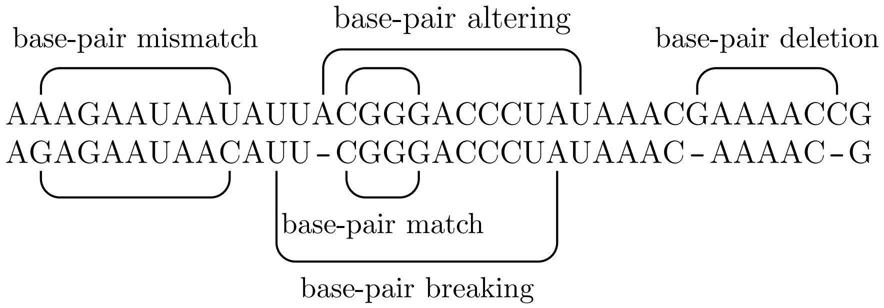


Figure 2.11: *Structural edit operations of Jiang et al.'s general edit model for RNA structures. Sequence edit operations that do not involve base-pairs are omitted in this figure.*

A General Edit Model for RNA Structures

Jiang et al. proposed a set of edit operations for RNA structures that are motivated by the evolution of structural RNA [94].

Edit operations An edit operation that affects the primary and the secondary structure transforms an RNA structure (S_1, P_1) into a structure (S_2, P_2) by modifying both, S_1 and P_1. Since a deletion or insertion of a base in S_1 requires to "adjust" the indexes of the base-pairs in P_1, the definition of edit operations is intricate on that level. I introduce a terminology for structural edit operations that is consistent with the terminology of the sequence and tree edit model. To uniquely define structural edit operations, the positions that are affected by the operation must be specified as well as the new base for base-replacements. For convenience, I define the rules in terms of their effect on sequence and structure. The parameterized edit operations can be derived from this description. Let be $u, v, w \in \Sigma^*_{\text{RNA}}$ and $a, b, c, d \in \Sigma_{\text{RNA}}$. Let the concatenated string $u'v'w'$ be a dot-bracket sequence in spirit of the Vienna strings that defines an RNA structure. Moreover, let the brackets "(" and ")" uniquely identify a base-pair. Note, the unique correspondence of a bracket string to an RNA structure requires different pairs of brackets in the presence of tertiary interactions. The symbol "." denotes an unpaired base. I arrange structure and sequence such that the structure is shown on top of the sequence. The changes by an edit operation are indicated as bold characters.

A family of structural conserved RNA molecules does often exhibit compensatory base mutations in stem regions. The replacement of a base-pair is modeled by the following edit operation:

$$\begin{array}{ccccc} u' & (& v' &) & w' \\ u & \mathbf{a} & v & \mathbf{b} & w \end{array} \quad \mapsto \quad \begin{array}{ccccc} u' & (& v' &) & w' \\ u & \mathbf{c} & v & \mathbf{d} & w \end{array} \qquad \text{(base-pair replacement)}$$

This notation is read as follows: (S_1, P_1) is edited to (S_2, P_2) where $S_1 = uavbw$, $P_1 \cong u'(v)w'$, $S_2 = ucvdw'$, and $P_2 \cong u'(v')w$. The operator $\cong$ means that the lefthand set of base-pairs is compatible with the base-pair pattern given by the righthand string. If $a = c$ and $b = d$ then the operation is also referred to as a *base-pair match*, otherwise it is denoted a *base-pair mismatch*. The disappearance of a base-pair, i.e. two pairing bases are lost during evolution, is given by:

$$\begin{array}{ccccc} u' & (& v' &) & w' \\ u & \mathbf{a} & v & \mathbf{b} & w \end{array} \quad \mapsto \quad \begin{array}{ccc} u' & v' & w' \\ u & v & w \end{array} \qquad \text{(base-pair deletion)}$$

During the evolution of an RNA structure, it can happen that the bond between two bases becomes too weak due to mutations in other regions of the structure. Accordingly, the disappearance of a base-pair bond is among the structural edit operations:

$$\begin{array}{ccccc} u' & (& v' &) & w' \\ u & a & v & b & w \end{array} \quad \mapsto \quad \begin{array}{ccccc} u' & . & v' & . & w' \\ u & a & v & b & w \end{array} \qquad \text{(base-pair breaking)}$$

The scenario where a base-pair bond disappears because one of the pairing bases is deleted is modeled by either of the following two edit-operations.

$$\begin{array}{ccccc} u' & (& v' &) & w' \\ u & \mathbf{a} & v & b & w \end{array} \quad \mapsto \quad \begin{array}{cccc} u' & v' & . & w' \\ u & v & b & w \end{array} \qquad \text{(base-pair altering right)}$$

$$\begin{array}{ccccc} u' & (& v' &) & w' \\ u & a & v & \mathbf{b} & w \end{array} \quad \mapsto \quad \begin{array}{cccc} u' & . & v' & w' \\ u & a & v & w \end{array} \qquad \text{(base-pair altering left)}$$

Bases that are not paired undergo the classical sequence edit operations:

$$\begin{array}{ccc} u' & . & v' \\ u & \mathbf{a} & v \end{array} \quad \mapsto \quad \begin{array}{ccc} u' & . & v' \\ u & \mathbf{c} & v \end{array} \qquad \text{(base-replacement)}$$

$$\begin{array}{ccc} u' & . & v' \\ u & \mathbf{a} & v \end{array} \quad \mapsto \quad \begin{array}{cc} u' & v' \\ u & v \end{array} \qquad \text{(base-deletion)}$$

Each of the edit operations can also be read and applied from right to left. For edit operations that involve the deletion of bases or base-pairs this defines the corresponding insert versions. Figure 2.11 shows the edit operations in an alignment on the sequence and structure level.

The concept of edit-sequences can be naturally applied: Let E be an edit-sequence $e_1, e_2, \ldots, e_n$. E transforms (S, P) into (S', P') if there is a sequence of structures $(S_0, P_0), (S_1, P_1), \ldots, (S_n, P_n)$ such that $(S, P) = (S_0, P_0)$, $(S', P') = (S_n, P_n)$ and (S_i, P_i) results from the application of e_i to (S_{i-1}, P_{i-1}) for $i \in [1, n]$. Let δ be a cost function defined on edit operations. The cost of an edit-sequence E is the sum of costs of its edit operations, that is: $\delta(E) = \sum_{i=1}^{n} \delta(e_i)$. The *general edit distance* δ_{GE} between structures (S_1, P_1) and (S_2, P_2) is the minimum cost that is necessary to transform (S_1, P_1) into (S_2, P_2). Formally,

$$\begin{aligned} \delta_{\mathrm{GE}}((S_1, P_1), (S_2, P_2)) = \min\{\delta(E) \mid E \text{ is an edit sequence} \\ \text{transforming } (S_1, P_1) \text{ into } (S_2, P_2)\}. \end{aligned} \tag{2.17}$$

Algorithms Jiang et al. provided algorithms and complexity results for a fixed scoring scheme, i.e. the cost of an edit operation does not account for the involved bases, or equivalently, it is a constant [94]. Computing δ_{GE} between (S_1, P_1) and (S_2, P_2) where P_1 is a tertiary structure and $P_2 = \emptyset$ is MAX SNP-hard. For a restricted model that omits the base-pair altering and base-pair deletion edit operations, they propose an algorithm that requires $O(|S_1|^2 \cdot |S_2|^2)$ time. If P_1 is a secondary structure and $P_2 = \emptyset$ the general (unrestricted) problem is solvable in $O(|S_1| \cdot |S_2|)$ time. The case when both P_1 and P_2 are secondary structures is not considered in [94]. I will show in Section 2.5.5 that the the general edit model with a certain scoring function is NP-hard.

Bafna et al.'s Model

Bafna et al. introduced a sequence alignment problem for RNA secondary structures that maximizes both, base and base-pair replacement scores [4]. Let $\alpha(a, b)$ be the score for replacing base a by base b and let $\beta(a{\circ}b, c{\circ}d)$ be the score for relabeling a base-pair $a{\circ}b$ by base-pair $c{\circ}d$. Given an alignment A of sequences S_1 and S_2, I define A_{S_i} to be the ith row in A. Let $gap_{S_i}[j]$ be the number of gaps that are inserted in S_i up to the jth position in A. Formally:

$$gap_{S_i}[j] = \begin{cases} j & \text{if } A_{S_i}[j] = \text{'}\lambda\text{'}, \\ |\{l \mid A_{S_i}[l] = \text{'}\lambda\text{' and } l \leq j\}| & \text{otherwise.} \end{cases}$$

Bafna et al. do the following trick to for a compact definition of their model: They define $S_i[0] =' \lambda'$. If there is a gap in S_1 at position i, $S_1[i - gap_{S_1}[i]]$ evaluates to "λ" which corresponds to an insertion. The corresponding holds for S_2. Let m be the number of columns in an alignment A. The score of A is the sum of scores of the aligned bases, be they paired or unpaired, and the scores of the aligned base-pairs. The sequence score α is defined as

$$\alpha(A) = \sum_{1 \leq i \leq m} \alpha(S_1[i - gap_{S_1}[i]], S_2[i - gap_{S_2}[i]]).$$

The base-pair scoring is defined as:

$$\beta(A) = \sum_{1 \le i \le j \le m} \beta(S_1[i - gap_{S_1}[i]] \circ S_1[j - gap_{S_1}[j]], S_2[i - gap_{S_2}[i]] \circ S_2[j - gap_{S_2}[j]])$$
$$\text{where } (i - gap_{S_1}[i], j - gap_{S_1}[j]) \in P_1$$
$$\text{and } (i - gap_{S_2}[i], j - gap_{S_2}[j]) \in P_2.$$

Bafna et al.'s score σ_{BAF} is the sum of these scores:

$$\sigma_{\mathrm{BAF}}(A) = \alpha(A) + \beta(A) \tag{2.18}$$

The similarity score of secondary structures (S_1, P_1) and (S_2, P_2) is then given by:

$$\sigma_{\mathrm{BAF}}((S_1, P_1), (S_2, P_2)) = \max\{\sigma_{\mathrm{BAF}}(A) \mid A \text{ is an alignment of } S_1 \text{ and } S_2\} \tag{2.19}$$

Note that S_1 and S_2 are sequences and, thus, A is a sequence alignment.

Algorithms Bafna et al. provide an algorithm that computes $\sigma_{\mathrm{BAF}}((S_1, P_1), (S_2, P_2))$ in $O(|S_1|^2 \cdot |S_2|^2)$.

Bafna et al.'s Model Revisited Bafna et al.'s model has been criticized for not systematically treating base-pairs as basic units [45, 94]. I show that their model can be expressed in the general edit model with a special scoring scheme: Function α scores base replacements, base-insertions and base-deletions. The scoring contributions are $\alpha(a, b), \alpha(\lambda, b)$ and $\alpha(a, \lambda)$, respectively. Clearly, function β in Equation (2.18) does only account for base-pair replacements. In this case, the function α contributes additionally to the overall score for the aligned base-pairs. Thus, the score for a base-pair replacement of $a \circ b$ with $c \circ d$ is $\beta(a \circ b, c \circ d) + \alpha(a, c) + \alpha(b, d)$. Otherwise, a base, be it paired or unpaired, can be aligned with any other base and the scoring contributions for aligning a base a with a base b is $\alpha(a, b)$. A scoring contribution of 0 for the base-pair breaking operation allows to align paired bases to unpaired bases without a penalty. The deletion of a base-pair is composed of a base-pair breaking and two base-deletions. The corresponding holds for the base-pair insertion. A base-pair altering is composed of a base-pair breaking, a base-match and a base-indel. Summarizing these observations, σ_{BAF} can be calculated by employing the following scoring scheme for Jiang et al.'s general edit model:

edit operation	score
base replacement	$\alpha(a, b)$
base indel	$\alpha(a, \lambda)$ and $\alpha(\lambda, b)$
base-pair replacement	$\beta(a{\circ}b, c{\circ}d) + \alpha(a, c) + \alpha(b, d)$
base-pair breaking	0

I conclude that Bafna et al.'s model is a proper structural alignment model which means that it can be expressed in Jiang et al.'s general edit model. Whether the scoring of edit operations is a good choice or not remains to be analyzed.

The Longest Arc-Preserving Common Subsequence Problem

The longest arc-preserving common subsequence problem is an extension of the classic longest common subsequence problem. A sequence S' is a *subsequence* of a sequence S if S' can be obtained from S by deleting characters. Given a set of sequences $S_1, S_2, \ldots, S_n$, the *longest common subsequence problem* asks for the longest sequence S' that is a subsequence of $S_1, S_2, \ldots, S_n$.

Mostly driven by the application of RNA structure comparison, including tertiary structures, Evans generalized the problem for arc-annotated sequences [45]. Let (S_1, P_1) and (S_2, P_2) be arc annotated sequences which means that P_1 and P_2 can be tertiary structures throughout this section. A longest common subsequence S' of S_1 and S_2 induces a mapping between characters in S_1 and S_2 by associating the characters i_k in S_1 and j_k in S_2, that correspond to the kth position of S'. Suppose $M = \{(i_1, j_1), (i_2, j_2), \ldots, (i_{|S'|}, j_{|S'|})\}$ is such a mapping. The longest common subsequence S' is *arc-preserving* if the arcs touched by the mapping are preserved. That is, for any $(i_k, j_k), (i_l, j_l) \in M$ holds:

$$(i_k, i_l) \in P_1 \quad \textit{iff} \quad (j_k, j_l) \in P_2. \tag{2.20}$$

The *longest arc-preserving common subsequence (LAPCS) problem* is to find a longest common subsequence S' that is arc-preserving.

Different instances of the problem, depending on the complexity of the arc set (here the complexity of RNA structures), are studied in the literature. The relevant instances in the context of RNA sequence and structure comparison are LAPCS(P_1, P_2) where P_i belongs to one of the following classes:

- PLAIN: no structure, i.e. $P_i = \emptyset$
- NESTED: P_i is a secondary structure
- CROSSING: P_i is a tertiary structure

I follow this terminology since it is established in the literature concerning LAPCS problems [2, 45, 93, 114]. I review the most important results and comment on the LAPCS(NESTED,NESTED) problem which is particularly interesting for comparing RNA secondary structures in the following.

Algorithms LAPCS(PLAIN,PLAIN) is the well known longest common subsequence problem which can be solved in $O(|S_1| \cdot |S_2|)$ [76]. If the number of sequences is unrestricted this problem is NP-complete [124]. Otherwise, if at least one structure is CROSSING, the problem is NP-hard [45]. A maximization optimization problem, such as the LAPCS problem, is *α-approximable* if there exists a polynomial time algorithm $\mathcal{A}$ and a positive number α such that the output of $\mathcal{A}$ is within a factor $\frac{1}{\alpha}$ of the optimum. If at least one structure is CROSSING. the LAPCS problem is also MAX SNP-hard which has the consequence that it is not approximable within $\alpha = 1 + \epsilon$ for some positive ϵ [93]. A 2-approximation algorithm for these problems is proposed in [93].

The probably most relevant problem in the context of RNA structures is the LAPCS(NESTED,NESTED) problem to compare RNA secondary structures. The NP-hardness of this problem was shown in [114].
A LAPCS(NESTED,NESTED) that can be obtained by at most k_1 and k_2 character deletions (together with the corresponding arcs) can be calculated in $O(3.31^{k_1+k_2})$ [2]. A polynomial time algorithm for the LAPCS(NESTED, PLAIN) problem, running in $O(|S_1| \cdot |S_2|^3)$ time, is presented in [93].

LAPCS(NESTED,NESTED) Revisited A longest arc-preserving common subsequence of secondary structures (S_1, P_1) and (S_2, P_2) maps characters from S_1 to S_2. In the following, I observe which edit operations of the general edit model are compatible with such a mapping, resulting in an equivalent edit based description of the LAPCS(NESTED,NESTED) problem. The arc-preserving property (2.20) of a longest arc-preserving common subsequence guarantees that if both bases of a base-pair are mapped, then they must be mapped to bases that are also paired. In terms of the general edit model for RNA structures this means that there must exist a base-pair match operation but no base-pair breaking. The base-pair match adds two new characters to the longest arc-preserving common subsequence. The base-pair breaking operation can be excluded by assigning an infinite negative score to it. If only one base of a base-pair is mapped, then the other base must not exist in the mapping. This adds one new character to the longest arc-preserving common subsequence. The arc-altering operations model exactly this scenario. Clearly, a base-pair deletion, i.e. both partners and the connecting arc are deleted, is also compatible with a LAPCS mapping. If a character is not paired, it can be mapped (matched) to another unpaired base (the mapping to a paired base is treated by the base-pair altering function) or not appear in the mapping. The sequence edit operations base-match and base-indel handle these cases. Clearly, a longest arc-preserving common subsequence does not allow any mismatches and, hence, the scoring contribution for those cases must be $-\infty$. Summarizing these observations, the length of a LAPCS can be calculated in Jiang et al.'s general edit model using the following scoring scheme:

edit operation	score
base match	1
base mismatch	$-\infty$
base indel	0
base-pair match	2
base-pair mismatch	$-\infty$
base-pair indel	0
base-pair breaking	$-\infty$
base-pair altering	1

The LAPCS can be derived from the resulting alignment. The complexity of the LAPCS(NESTED,NESTED) problem was an important question until Lin et al. proved it to be NP-hard [114]. Since the computation of the general edit distance using the above scores solves the LAPCS problem, I conclude that the computation of the general edit distance for RNA secondary structures is a NP-hard problem for the above scoring scheme. I assume that the complexity results from the presence of the base-pair altering operations. If those must be considered explicitly, i.e. the score is not build from simpler edit operations, the number of resulting subproblems grows exponentially. This remains to be further analyzed.

Zhang et al.'s Model

Zhang et al. considered RNA secondary structure trees in the natural representation that are compared under the tree edit and alignment model in [238]. The entities of the tree nodes are bases and base-pairs (see Section 2.3). Thus, the classic edit operations replace, insert and delete can be applied to either an unpaired base or a base-pair. A replacement of a base by a base-pair is prohibited. Ma et al. extended this model for general RNA structures by extending the mapping concept of the tree edit model for general RNA structures which is the central definition of this line of work [123, 222, 242]. The essential extension of the mapping is a new condition for "crossing" base-pairs. Intuitively, the crossing pattern of tertiary interactions should be conserved. I do not go into the details of their mapping definitions, since their model was constructed on the assumption of certain edit operations on structures. I will revisit their models in terms of Jiang et al.'s general model.

Algorithms Computing $\delta_{\mathrm{ZHA}}((S_1, P_1), (S_2, P_2))$ where P_1 and P_2 are tertiary structures is MAX-SNP hard [123]. Ma et al. considered a simpler edit model for tertiary structures which restricts mappings between tertiary structures to preserve secondary structure. Essentially, their algorithm deletes tertiary structure interactions such that the resulting secondary structure alignment is optimal. Let $\mathit{stem}(P)$ be the number of stacking regions (stems) in an RNA structure (S, P). Their algorithm requires $O(\mathit{stem}(P_1) \cdot \mathit{stem}(P_2) \cdot |S_1| \cdot |S_2|)$ time and $O(\mathit{stem}(P_1) \cdot \mathit{stem}(P_2))$ space. Collins et al. presented a variant of δ_{ZHA} with the constraint that bases and base-pairs can be specified that must be replaced by each other. They do not improve the complexity, but their technique reduces the search space and consequently the runtime [29]. Moreover, they propose a two step strategy for tertiary structures: In the first step, tertiary structures are ignored resulting in a secondary structure alignment. In the second step, the secondary structure alignment is used to restrict the tertiary structure alignment.

Zhang et al.'s Model Revisited The edit operations in Zhang et al.'s edit model can be applied to either unpaired bases or base-pairs. According to Jiang et al.'s model the structural edit operations are: base-pair replace and base-pair indel. The sequence counterparts are the operations base-replace and base-indel. An edit operation that works on both unpaired base and a base-pair is not defined in their model. Thus, there is no base-pair altering and base-pair breaking operation. An infinite negative score for these edit operations is sufficient to calculate Zhang et al.'s model under the general edit model for RNA structures:

edit operation	score
base match	α_m
base mismatch	α_{mm}
base indel	α_{id}
base-pair match	β_m
base-pair mismatch	β_{mm}
base-pair indel	β_{id}
base-pair breaking	$-\infty$
base-pair altering	$-\infty$

2.5.6 Graphs

The most general mathematical construct to model relations between certain objects is a graph. Clearly, an RNA tertiary structure can be modeled as a graph where the vertices are bases and the edges are interactions between them. Note, this concerns topological rather than geometric aspects of RNA molecules. For example, such a graph abstracts from the relative angles between stems.

Edit Models

Wan et al. considered the generalization of the tree edit model for graphs, these are approximate graph isomorphism and subgraph isomorphism [218]. Both are known to be NP-complete. They outline an application where RNA structures (not restricted to secondary structures) are compared under this model.

Eigenvalue Spectrum of the Laplacian Matrix

In the Schlicks's group two simpler types of graphs are considered [47, 52, 53]. The one are tree graphs, corresponding to a collapsed form of the natural tree representation (see Section 2.3), where collapsed means that connected non-branching nodes are merged to one node (ignoring labels). The other is *dual graphs*. A dual graph can represent all tree like RNA structures as well as pseudoknotted structures. They focus on the problem of quantitatively characterizing known structural motifs to identify missing or favored motif topologies. For the topological classification of structures they consider the eigenvalue spectrum of the Laplacian matrix obtained from the graph's adjacency matrix. In particular, the second eigenvalue reflects the overall pattern of connectivity for a graph. Barash used the second eigenvalue to detect structural changes in RNA that are caused by single point mutations [6].

2.6 Discussion

The multitude of structure comparison models presented in Section 2.5 gives rise to the question why this thesis presents another RNA structure comparison model. Otherwise, this shows that RNA structure comparison is an active research field and the problem is not sufficiently solved.

Nowadays, the detection of locally structure conserved motifs in RNA molecules is a hot topic in molecular biology. On the algorithmic side, the problem of finding local similar structures, given RNA secondary structures, has not been studied thoroughly. The *similar consensus problem* for trees is the only contribution, I am aware of, to detect local similar regions in RNA secondary structures (see 2.5.4). However, this model calculates distance instead of similarity. As the distance between equal substructures is always zero, the size of substructures must be considered additionally. Hence, in the similar consensus problem, the largest subtree within some distance threshold is sought. A similarity version in spirit of the *Smith-Waterman algorithm* [184] for trees would be more convenient to calculate local similar structures. Moreover, the similar consensus problem consideres subtrees. This is too restrictive since neighboring subtrees should be considered as local structures as well, i.e. two adjacent stems in a multiloop could be the most similar substructure which corresponds to two different subtrees.

Another problem that has not been addressed thoroughly is the problem of comparing multiple RNA secondary structures. As multiple sequence alignments emphasize sequence conserved regions, multiple structure alignments emphasize structural conserved regions. A multiple structural alignment is useful for phylogenetic analyses, identification of conserved motifs, and domain and structure prediction.

In the follwowing, I motivate my choice of the tree alignment model to address the above problems. The model that I consider should have the following properties:

- a biologically reasonable edit model,
- suitable for a generalization to multiple structures,
- build upon an adequate data structure for local similarity problems,
- allow algorithms with a low computational complexity.

Base-pair distances are suitable to compare structures that have the same length, i.e. the same number of nucleotides. If the structures to be compared have a different length, edit based approaches provide a better distance measure.

The approach to apply classical sequence alignments to string representations of RNA secondary structures is more a historical remark. At the time these were invented, structural alignment strategies were just about to emerge. More elaborate models are edit and alignment models for trees and arc-annotated sequences.

Unlike sequences, trees are convenient to express substructures of RNA secondary structures as coherent parts of the data structure. In explanation, adjacent base-pairs are neighbored in a tree while in a sequence they are split in the $5'$ and $3'$ bases connected by arcs. From the viewpoint of being able to generalize the model to align multiple structures, the tree alignment model has an interesting property. Alignments of trees are trees. Thus, a tree alignments can, again, be aligned in the tree alignment model. This makes virtually every progressive strategy known for the calculation of multiple sequence alignments applicable to the calculation of multiple tree alignments. Another property of tree based approaches is that the chosen tree representation can control the level of abstraction of RNA secondary structures. In the end, the time complexity for calculating tree alignments meets practical requirements.

Chapter 3

Algorithms for Global and Local Forest Similarity

The tree alignment distance and a dynamic programming algorithm that calculate this distance was introduced by Jiang et al. [95]. In this chapter, I extend Jiang et al.'s tree alignment model to forests. Unlike Jiang et al., I consider the similarity version of the alignment problem and introduce new local similarity variants. A uniform, purely forest based notation makes, as I believe, the understanding of the concepts easier. I systematically identify the subproblems that must be considered to get an overall solution. Based on these observations, I provide an efficient tabulation technique for intermediate results. The resulting algorithms are more compact and, hence, easier and faster to implement. From a practical viewpoint, RNA secondary structures have a forest structure in general and the introduction of a virtual root node requires special cases for the application of a tree distance. In particular, the scoring function must guarantee to match the virtual root nodes and in the structural alignment these must be omitted. It is much more convenient to compare forests directly. I have published central ideas of this Chapter in [77].

3.1 Preliminaries

Recall that a forest F is a sequence of trees. Let $len(F)$ be the number of trees in F, i.e. the length of sequence F. Let $i{:}F$ be the forest consisting of the first i trees of F (*prefix*), while $F{:}j$ is the forest consisting of the last j trees of F (*suffix*). $_{\mathrm{i}}]F[_{\mathrm{j}}$ denotes the forest F without the prefix $i{:}F$ and the suffix $F{:}j$ (*subword*). I use this notation, since the identification of a subword as a prefix of a suffix, e.g. $i{:}F{:}j$, could also be read as a suffix of a prefix, which is ambiguous without introducing brackets. For each node v in F, $pre_F(v)$ is the index of v according to the left-to-right pre-order traversal of F. I define $F[i]$ to identify a node by its index, i.e. $F[pre_F(v)] = v$. If F is not the empty forest, $F_\perp$ is the root node of the first tree in F, that is $F[1]$. I define $F^\downarrow$ to be the forest consisting of the children trees of $F_\perp$ and $F^\rightarrow = F{:}(len(F) - 1)$ to be the forest of the right sibling trees of $F_\perp$. Note that $F^\downarrow$ and $F^\rightarrow$ can be empty forests. Throughout this section, I refer to the two forests that are aligned as F and G.

3.2 Alignment of Forests

An alignment of trees is a tree. Following the general concept of an alignment (see Section 2.5.3), an alignment of forests is a forest. The tree alignment definition is generalized straightforward to forests as follows:

$$A \in \mathcal{F}(\Sigma_\lambda^2) \textit{ is an alignment of forests } F, G \in \mathcal{F}(\Sigma) \text{ iff}$$
$$F = \pi(A|_1) \text{ and } G = \pi(A|_2). \tag{3.1}$$

I consider the similarity version of the forest alignment which is important to define local similarity variants of the problem (This will be further explained in Section 3.4). The *alignment similarity* σ_{FA} of forests F_1 and F_2 is the maximum score that an alignment of F_1 and F_2 can achieve. That is:

$$\sigma_{\mathrm{FA}}(F, G) = \max\{\sigma(A) \mid \text{A is an alignment of } F \text{ and } G\}. \tag{3.2}$$

3.3 A Global Forest Alignment Algorithm

Jiang et al. presented an algorithm for the calculation of the tree alignment distance which has the best known worst case complexity $O(|T_1| \cdot |T_2| \cdot (\mathit{degree}(T_1) + \mathit{Degree}(T_2))^2)$ [95].

The recursive nature of forests leads naturally to dynamic programming algorithms. This sort of algorithms is structurally recursive and avoid recalculation of the same subproblem by tabulating intermediate results. Adhering to the principles of *algebraic dynamic programming* [56, 58], I consider the search space of a problem (all possible alignments) and its evaluation (e.g. scoring, counting) separately. To derive a dynamic programming algorithm from the search space observations, two question must be answered:

1. Which subproblems arise in the recursion scheme?

2. What is the order of calculation?

The answer to the first question identifies the *relevant subforests* of the problem. Thereupon, an index based notation that is necessary for an implementation based on matrix recurrences can be derived. The answer to the second question is important to formulate an imperative algorithm. Clearly, everything must be calculated before it is used[1].

3.3.1 The Search Space of Forest Alignments

To calculate similarity of forests, all their alignments must be considered. This set is the *search space.* The enumeration of all possible alignments of two forests can be done in a structurally recursive fashion. Suppose A is an alignment of F and G. Depending on $\mathit{label}(A_\perp)$, the possible forests $A^\downarrow$ and $A^\rightarrow$ are determined. The following case analysis is based on Definition (3.1).

[1] The principle of *referential transparency* makes this obsolete in functional programming languages like *Haskell* [155] which is exploited in Giegerich's algebraic dynamic programming approach [56].

Lemma 3.1. *Let A be an alignment of $F, G \in \mathcal{F}(\Sigma)$. If F or G are empty forests, A is either the empty forest, or its labels are solely deletions or solely insertions. If F and G are both non-empty forests, then label$(A_\perp)$ is of the form $(a, b), (\lambda, b)$ or (a, λ) for some $a, b \in \Sigma$. This leads to the following case distinction:*

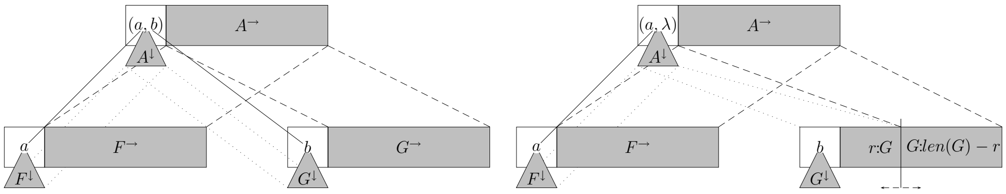
1. *If label$(A_\perp) = (a, b)$, then the following is true:*
   - *$a =$ label$(F_\perp)$ and $b =$ label$(G_\perp)$,*
   - *$A^\downarrow$ is an alignment of $F^\downarrow$ and $G^\downarrow$ and $A^\rightarrow$ is an alignment of $F^\rightarrow$ and $G^\rightarrow$.*

2. *If label$(A_\perp) = (a, \lambda)$, then the following is true:*
   - *$a =$ label$(F_\perp)$,*
   - *for some $r \in [0, len(G)]$, $A^\downarrow$ is an alignment of $F^\downarrow$ and $r{:}G$ and $A^\rightarrow$ is an alignment of $F^\rightarrow$ and $G{:}(len(G) - r)$.*

3. *If label$(A_\perp) = (\lambda, b)$, then the following is true:*
   - *$b =$ label$(G_\perp)$,*
   - *for some $r \in [0, len(F)]$, $A^\downarrow$ is an alignment of $r{:}F$ and $G^\downarrow$ and $A^\rightarrow$ is an alignment of $F{:}(len(F) - r)$ and $G^\rightarrow$.*

Proof. Follows directly from Definition (3.1) and the definition of function π in Equation (2.10). □

Figure 3.1 gives a graphical view of Lemma 3.1. The search space of all possible alignments of F and G is determined by the Cases 1-3, and by all possible choices of split position r in Cases 2 and 3. Scoring the alignments of the search space follows the same structural recursive pattern. The similarity of F and G is the maximum of the scores $\sigma(a, b)$, $\sigma(a, \lambda)$ and $\sigma(\lambda, b)$, each added to the similarity scores of the appropriate subforests. Figure 3.2 shows the recursive formula for the calculation of the forest alignment similarity that



(a) Case 1

(b) Case 2

Figure 3.1: *Illustration of Case 1 and Case 2 of Lemma 3.1. The shaded triangle symbolizes $F^{\downarrow}$ and the shaded rectangle symbolizes $F^{\rightarrow}$. The prefix/suffix pairs of G are indicated by the vertical line "splitting" G.*

follows directly from this observation. Clearly, Bellman's principle of optimality is satisfied [9]. To turn our case analysis into a dynamic programming algorithm, intermediate results must be tabulated.

3.3.2 Implementation based on Matrix Recurrences

The key notion for forest alignment problems (and also for the tree alignment model) is the closed subforest:

$$\text{A consecutive sequence } T_1, \ldots, T_n \text{ of sibling trees in } F \text{ is } \textit{a closed subforest } (\textit{csf}) \text{ of } F. \quad (3.3)$$

A *csf* F' of F is *maximal* if it cannot be extended to the left or right, formally, there is no *csf* F'' of F such that F' is a proper prefix or suffix of F''. Clearly, the empty forest $\emptyset$ and forest F itself are *csfs* of F. It is easy to see that if F' is a *csf* of F and F'' is a *csf* of F', then F'' is a *csf* of F (*closed subforest transitivity*). The pairs of subforests that actually arise in the recursive calculation of the tree alignment similarity, the relevant subforests, are subject of the following Lemma.

Lemma 3.2. *Let $\bar{F}$ and $\bar{G}$ be maximal closed subforests of F and G, respectively. The pairs of subforests that are relevant for the calculation of*

$$\begin{aligned}
relabel(F,G) &= \delta(label(F_\perp) \to label(G_\perp)) + \sigma_{\mathrm{FA}}(F^\downarrow, G^\downarrow) + \sigma_{\mathrm{FA}}(F^\to, G^\to)\\
delete(F,G) &= \delta(label(F_\perp) \to \lambda) + \max_{r\in[0,len(G)]} \left\{\sigma_{\mathrm{FA}}(F^\downarrow, r{:}G) + \sigma_{\mathrm{FA}}(F^\to, G{:}(len(G)-r))\right\}\\
insert(F,G) &= \delta(\lambda \to label(G_\perp)) + \max_{r\in[0,len(F)]} \left\{\sigma_{\mathrm{FA}}(r{:}F, G^\downarrow) + \sigma_{\mathrm{FA}}(F{:}(len(F)-r), G^\to)\right\}
\end{aligned}$$

$$\sigma_{\mathrm{FA}}(F,G) = \begin{cases}
0 & \text{if } F=\emptyset \text{ and } G=\emptyset\\
\delta(label(F_\perp)\to\lambda) + \sigma_{\mathrm{FA}}(F^\downarrow,\emptyset) + \sigma_{\mathrm{FA}}(F^\to,\emptyset) & \text{if } F\neq\emptyset \text{ and } G=\emptyset\\
\delta(\lambda\to label(G_\perp)) + \sigma_{\mathrm{FA}}(\emptyset,G^\downarrow) + \sigma_{\mathrm{FA}}(\emptyset,G^\to) & \text{if } F=\emptyset \text{ and } G\neq\emptyset\\
\max\begin{cases} relabel(F,G)\\ delete(F,G)\\ insert(F,G)\end{cases} & \text{otherwise}
\end{cases}$$

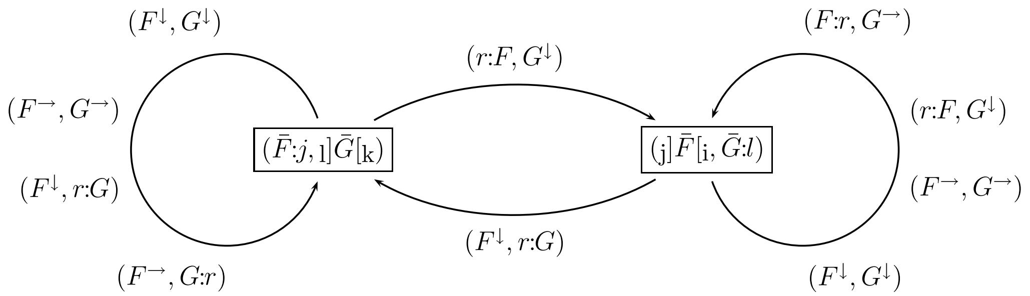
Figure 3.2: *Recursive function to calculate the forest alignment similarity σ_{FA} of forest F and G.*

$\sigma_{FA}(F,G)$ due to Figure 3.2 have the form $(\bar{F}{:}j, {}_k]\bar{G}[_l)$ and $({}_j]\bar{F}[_i, \bar{G}{:}l)$.

Proof. Both pairs of *csfs* $(\bar{F}{:}j, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{i}}]\bar{F}[_{\mathrm{j}}, \bar{G}{:}l)$ where each of i,j,k,l equals 0 represent the *csf* pair $(\bar{F},\bar{G})$. I consider all possible transitions to subforests due to Lemma 3.1. These are:

- Case 1: $(F^\downarrow, G^\downarrow), (F^\to, G^\to)$,
- Case 2: $(F^\downarrow, k{:}G), (F^\to, G{:}l)$,
- Case 3: $(i{:}F, G^\downarrow), (F{:}j, G^\to)$.

The following graph shows the pairs of *csfs* $(\bar{F} : j, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{j}}]\bar{F}[_{\mathrm{i}}, \bar{G} : l)$ surrounded by boxes. The arrows indicate possible transitions targeting to the index pair that is sufficient to represent the result of the transition.


$(F^{\downarrow}, G^{\downarrow})$
$(F^{\rightarrow}, G^{\rightarrow})$
$(F^{\downarrow}, r{:}G)$
$(F^{\rightarrow}, G{:}r)$
$(\bar{F}{:}j, {}_{\mathrm{l}}]\bar{G}[_{\mathrm{k}})$
$(r{:}F, G^{\downarrow})$
$(F^{\downarrow}, r{:}G)$
$({}_{\mathrm{j}}]\bar{F}[_{\mathrm{i}}, \bar{G}{:}l)$
$(F{:}r, G^{\rightarrow})$
$(r{:}F, G^{\downarrow})$
$(F^{\rightarrow}, G^{\rightarrow})$
$(F^{\downarrow}, G^{\downarrow})$

Each transition results in pairs of closed subforests that can be expressed in one of the two forms. Thus, the subforests $(\bar{F}{:}j, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{i}}]\bar{F}[_{\mathrm{j}}, \bar{G}{:}l)$ are sufficient to describe the relevant subforests. For all combinations of i, j, k, l, the subforests $(\bar{F}{:}j, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{i}}]\bar{F}[_{\mathrm{j}}, \bar{G}{:}l)$ can be reached from the pair (F, G) by a series of transitions. Hence, Lemma 3.2 describes exacltly the relevant subforests. □

It is obvious that each relevant subforest is a closed subforest. Note that the converse does not hold, i.e. the pair of *csfs* $({}_{\mathrm{i}}]F[_{\mathrm{j}}, {}_{\mathrm{k}}]G[_{\mathrm{l}})$ where each of i, j, k, l is greater than 0 is not a relevant pair of subforests for the calculation of $\sigma_{\mathrm{FA}}(F, G)$.

Tabulation

For a transparent description of my algorithms, I use a two stage mapping $\beta_F \cdot \alpha_F$. The function α_F provides a mapping from *csfs* of F to index pairs and allows for efficient transitions from a *csf* to its relevant subforests. The function β_F maps these index pairs to linear table indices. In this way, I reduce table dimension and space consumption in practice. For any non-empty *csf* F' of F, I define

$$\alpha_F(F') = (pre_F({F'}_{\perp}), len(F')). \tag{3.4}$$

The empty forest is represented ambiguously by any index pair $(i, 0)$. If (i, j) is an index pair representing a *csf*, then i is called the *node index* and j the *length index*.

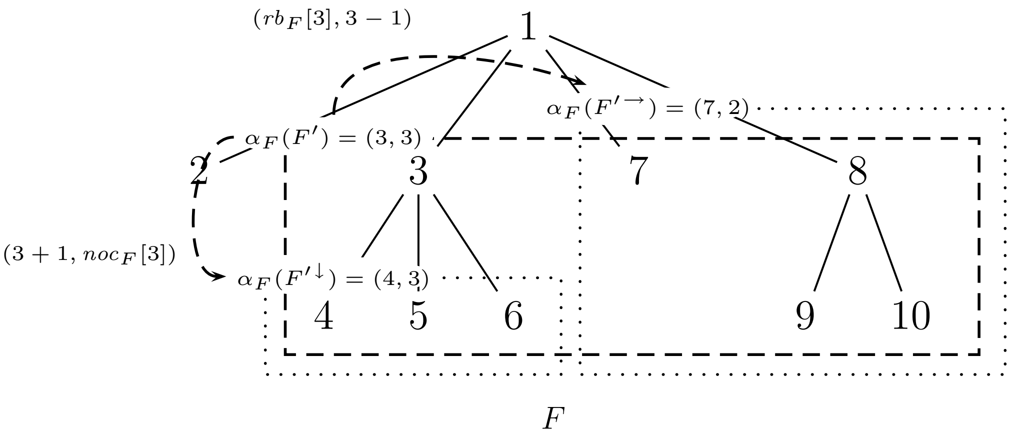

$(rb_F[3], 3-1)$
1
$\alpha_F(F'^{\rightarrow}) = (7,2)$
$\alpha_F(F') = (3,3)$
2
3
7
8
$(3+1, noc_F[3])$
$\alpha_F(F'^{\downarrow}) = (4,3)$
4
5
6
9
10
F

Figure 3.3: *This figure illustrates the closed subforest index pair representation. The transitions of the* csf F' *to* $F'^{\downarrow}$ *and* $F'^{\rightarrow}$ *are indicated by the arrows which are annotated by the corresponding calculations involving the tables* noc_F *and* rb_F*.*

Let $noc_F[i]$ be the number of children of $F[i]$ and let $rb_F[i]$ be the preorder index of the right brother node of $F[i]$. If there is no right brother, then $rb_F[i] = 0$. If F' is a non-empty *csf* of F and $\alpha_F(F') = (i, j)$, then:

- $\alpha_F(F'^{\downarrow}) = (i+1, noc_F[i])$. If $F[i]$ is not a leaf, $i+1$ is the index of the leftmost child of $F[i]$ and $noc_F[i] = len(F'^{\downarrow})$. If $F[i]$ is a leaf, the resulting *csf* is the empty forest represented by $(i+1, noc_F[i]) = (i+1, 0)$.

- $\alpha_F(F'^{\rightarrow}) = (rb_F[i], j-1)$. If $F[i]$ has a right brother, this is quite obvious. Otherwise $j-1=0$ and the resulting forest is empty.

Clearly, $\alpha_F(F'^{\downarrow})$ and $\alpha_F(F'^{\rightarrow})$ can be computed in constant time, given $\alpha_F(F')$. Splitting F' into $r\!:\!F'$ and $F'\!:\!r$ yields to subforests represented by (i, r) and $(rb_F^r[i], j-r)$ where rb_F^r is the r-fold application of rb_F. Since the splits will be determined in order of increasing r, the amortized cost of each split is $O(1)$. Figure 3.3 illustrates the α-mapping and the index transitions on closed subforest-index pairs.

Now it is easy to derive matrix recurrences for a dynamic programming algorithm calculating forest alignment similarity. I just have to substitute the subforests in the formula in Figure 3.2 by the corresponding index pairs, and switch from enumeration of the search space to maximization of similarity. A four-dimensional matrix S^4_σ such that $S^4_\sigma(\alpha_F(F'), \alpha_G(G'))$ is the similarity of *csfs* F' and G' of F and G, respectively, would allow a straightforward tabulation. Such a tabulation technique requires $O(|F| \cdot \mathit{degree}(F) \cdot |G| \cdot \mathit{degree}(G))$ space. However, the four-dimensional tabulation wastes space for two reasons:

- The empty forest is represented ambiguously by all index pairs $(i, 0)$.
- Matrix S^4_σ is sparse. In explanation, let p be the number of siblings to the right of $F[i]$ plus one (including $F[i]$). For all $p < j \leq \mathit{degree}(F)$, (i, j) does not represent an existing *csf* of F. Hence, for these *csfs* (i, j), the matrix elements $S^4_\sigma((i, j), (k, l))$ are not used. The corresponding holds for *csfs* of G.

The concrete shape of the forests to be aligned determines the number of unused entries in S^4_σ. Even in the best case, when all internal nodes in the forests have the same out-degree p, nearly half of the table is not used. This becomes worse if the node degree varies. The second stage mapping β_F from indices pairs to one dimensional indices eliminates all unused entries[2]. For that purpose, an auxiliary table offset_F stores for each node index i the number of non-empty *csfs* having a node index less than i. The second stage mapping β_F is defined by

$$\beta_F(i, j) = \begin{cases} 0 & \text{if } j = 0 \\ \mathit{offset}_F[i] + j & \text{otherwise} \end{cases} \tag{3.5}$$

Table offset_F can be precomputed in $O(|F|)$ time and space. I define the right inverse β_F^{-1} of β_F by $\beta_F^{-1}(0) = (1, 0)$ and $\beta_F^{-1}(\beta_F(i, j)) = (i, j)$ for $\beta_F(i, j) \neq 0$.

I combine the previous ideas to give a dense tabulating algorithm calculating global forest alignment similarity. I compute matrix S_σ defined by

[2] It cannot eliminate entries that correspond to pairs of *csfs* that are not relevant due to Lemma 3.1.

$$
S_\sigma(x,y) = \begin{cases}
0 & \text{if } x = 0 \text{ and } y = 0 \qquad (1) \\[1ex]
\sigma(\mathit{lb}_F[i],\lambda) & \\
+S_\sigma(\beta_F(i+1,\mathit{noc}_F[i]),0)) & \\
+S_\sigma(\beta_F(\mathit{rb}_F[i],j-1),0) & \text{if } x > 0 \text{ and } y = 0 \qquad (2) \\[1ex]
\sigma(\lambda,\mathit{lb}_G[k]) & \\
+S_\sigma(0,\beta_G(k+1,\mathit{noc}_G[k])) & \\
+S_\sigma(0,\beta_G(\mathit{rb}_G[k],l-1)) & \text{if } x = 0 \text{ and } y > 0 \qquad (3) \\[1ex]
\max \begin{cases} \mathit{relabel}(x,y) \\ \mathit{delete}(x,y) \\ \mathit{insert}(x,y) \end{cases} & \text{otherwise} \qquad (4)
\end{cases}
$$

where $(i,j) = \beta_F^{-1}(x)$ and $(k,l) = \beta_G^{-1}(y)$

$$
\begin{aligned}
\mathit{relabel}(x,y) &= \sigma(\mathit{lb}_F[i],\mathit{lb}_G[k]) \\
&\quad +S_\sigma(\beta_F(i+1,\mathit{noc}_F[i]),\beta_G(k+1,\mathit{noc}_G[k])) \\
&\quad +S_\sigma(\beta_F(\mathit{rb}_F[i],j-1),\beta_G(\mathit{rb}_G[k],l-1)) \\
\mathit{delete}(x,y) &= \sigma(\mathit{lb}_F[i],\lambda) \\
&\quad + \max_{0\le r\le l} \left\{ \begin{array}{l} S_\sigma(\beta_F(i+1,\mathit{noc}_F[i]),\beta_G(k,r)) \\ +S_\sigma(\beta_F(\mathit{rb}_F[i],j-1),\beta_G(\mathit{rb}^r_G[k],l-r)) \end{array} \right\} \\
\mathit{insert}(x,y) &= \sigma(\lambda,\mathit{lb}_G[k]) \\
&\quad + \max_{0\le r\le j} \left\{ \begin{array}{l} S_\sigma(\beta_F(i,r),\beta_G(k+1,\mathit{noc}_G[k])) \\ +S_\sigma(\beta_F(\mathit{rb}^r_F[i],j-r),\beta_G(\mathit{rb}_G[k],l-1)) \end{array} \right\}
\end{aligned}
$$

Figure 3.4: *The recurrences for S_σ for computing the entries of S_σ. The Cases (1)-(3) of the recurrences involving empty forests are obvious. The similarity of two non-empty forests is determined by the maximum score for alignments A that have a relabeling, or a deletion, or an insertion at the root. The functions relabel, delete, and insert reflect the case distinction in Lemma 3.1.*

$$
S_\sigma(\beta_F(\alpha_F(F')),\beta_G(\alpha_G(G'))) = \sigma_{\mathrm{FA}}(F',G') \tag{3.6}
$$

for all *csfs* F' and G' of F and G, respectively. Since $\alpha_F(F) = (1,\mathit{len}(F))$ and $\alpha_G(G) = (1,\mathit{len}(G))$, the value in $S_\sigma(\beta_F(1,\mathit{len}(F)),\beta_G(1,\mathit{len}(G)))$ gives the similarity of F and G. The recurrences for S_σ are given in Figure 3.4.

The Order of Calculation

To complete the dynamic programming algorithm, the order of evaluating the entries in S_σ must be considered. Each element must be evaluated before it is used. Evaluating S_σ row by row or column by column, as done in the dynamic programming algorithm for sequence similarity [184], does not work here. Consider the data dependencies in the recurrences of Figure 3.4. Obviously, $S_\sigma(0,0)$ can be initialized to zero. If $\beta_F(i,j) > 0$, then $S_\sigma(\beta_F(i,j),0)$ depends on entries $S_\sigma(\beta_F(i+1, noc_F[i]),0)$ and $S_\sigma(\beta_F(rb_F[i], j-1),0)$. That is, either the node index strictly increases, or if $rb_F[i] = 0$, then $j = 1$ and hence $\beta_F(rb_F[i], j-1) = 0$. If $\beta_G(k,l) > 0$, then the corresponding holds for $S_\sigma(0, \beta_G(k,l))$. If $\beta_F(i,j) > 0$ and $\beta_G(k,l) > 0$ in the delete case (Lemma 3.1 Case 2), $S_\sigma(\beta_F(i,j), \beta_G(k,l))$ depends on $S_\sigma(\beta_F(i+1, noc_F[i]), \beta_G(k,r))$ and $S_\sigma(\beta_F(rb_F[i], j-1), \beta_G(rb^r_G[k], l-r))$ for some $r \in [0,l]$. Thus, either the node index increases strictly, or the length index decreases. The corresponding holds for the insert case (Case 3). Thus, S_σ can be evaluated in decreasing order of the node index and increasing order of the length index. This is done in Algorithm 3.1 that tabulates $gs_\sigma(F,G)$ for all relevant pairs of *csfs* F' and G' of F and G, while fulfilling the data dependencies that were just discussed. The iteration over the length index makes use of a table $\mathit{maxcsflen}_F$, defined by:

$$\mathit{maxcsflen}_F[i] = \max\{j \mid (i,j) \text{ is a } \mathit{csf} \text{ of } F\}. \tag{3.7}$$

The table $\mathit{maxcsflen}_F$ can be precomputed in $O(|F|)$ time and space, since

$$\mathit{maxcsflen}_F[i] = \begin{cases} 1 & \text{if } rb_F[i] = 0 \\ \mathit{maxcsflen}_F[i] = 1 + \mathit{maxcsflen}_F[rb_F[i]] & \text{otherwise.} \end{cases}$$

The corresponding holds for table $\mathit{maxcsflen}_G$. An optimal alignment can be obtained by backtracking. To facilitate this, the split position r should be stored with each optimal value resulting from a deletion or an insertion.

```
   **Input**: Forests $F$ and $G$, given by tables
          $lb_F, lb_G, noc_F, noc_G, rb_F, rb_G, \mathit{offset}_F, \mathit{offset}_G,$
          $\mathit{maxcsflen}_F, \mathit{maxcsflen}_G$
   **Output**: $\sigma_{\mathrm{FA}}(F, G)$ stored at
          $S_\sigma(\beta_F(i, \mathit{maxcsflen}_F[i]), \beta_G(k, \mathit{maxcsflen}_G[k]))$
1  $S_\sigma(0, 0) \leftarrow 0$
2  for $i \leftarrow |F|$ to $1$ do
3      for $j \leftarrow 1$ to $\mathit{maxcsflen}_F[i]$ do  Calculate $S_\sigma(\beta_F(i, j), 0)$
4  end
5  for $k \leftarrow |G|$ to $1$ do
6      for $l \leftarrow 1$ to $\mathit{maxcsflen}_G[k]$ do  Calculate $S_\sigma(0, \beta_G(k, l))$
7  end
8  for $i \leftarrow |F|$ to $1$ do
9      for $k \leftarrow |G|$ to $1$ do
10         for $j \leftarrow 1$ to $\mathit{maxcsflen}_F[i]$ do
11             Calculate $S_\sigma(\beta_F(i, j), \beta_G(k, \mathit{maxcsflen}_G[k]))$ as in Figure 3.4
12         end
13         for $l \leftarrow 1$ to $\mathit{maxcsflen}_G[k]$ do
14             Calculate $S_\sigma(\beta_F(i, \mathit{maxcsflen}_F[i]), \beta_G(k, l))$ as in Figure 3.4
15         end
16     end
17 end
```

Algorithm 3.1: Algorithm for the calculation of global forest alignment similarity σ_{FA}.

Efficiency Analysis

Time Efficiency According to the recurrences in Figure 3.4, each $S_\sigma(x, y)$ is calculated in $O(\mathit{degree}(F) + \mathit{degree}(G))$ time. Elements of table $\mathit{maxcsflen}_F$ and table $\mathit{maxcsflen}_G$ are bounded by $\mathit{degree}(F)$ and $\mathit{degree}(G)$, respectively. Thus, the initialization steps in Line 2-4 and Line 5-7 require $O(|F| \cdot \mathit{degree}(F)^2)$ and $O(|G| \cdot \mathit{degree}(G)^2)$ time. The main loop structure, ranging from Line 8 to 17, requires $O(|F| \cdot |G| \cdot (\mathit{degree}(F) \cdot (\mathit{degree}(F) + \mathit{degree}(G)) + \mathit{degree}(G) \cdot (\mathit{degree}(F) + \mathit{degree}(G)))$ time. After rearrangement of the terms, the overall time complexity of Algorithm 3.1 is $O(|F| \cdot |G| \cdot (\mathit{degree}(F) + \mathit{degree}(G))^2)$.

Space Efficiency The size of the table S_σ depends on the number of *csfs* in F and G. The following theorem gives an upper and lower bound for the number of closed subforests for forests with a fixed size and degree.

Theorem 3.1. *Let F be a forest with size n and degree k. The number of closed subforests in F, denoted by $csf(F)$, is bounded by:*

$$n + 1 \leq csf(F) \leq \frac{n \cdot (k-1)}{2} + 1$$

Proof. First, I consider the upper bound: Assume that $n = p \cdot k$ for some integer p, and there are p sequences $S_1, S_2, \ldots, S_p$ of sibling nodes each of length k. For each of the p sequences, there are $\sum_{i=1}^{k} i = \frac{k \cdot (k-1)}{2}$ different, non-empty *csfs* where the node index is an element of the sequence. For all $p = \frac{n}{k}$ sequences of sibling nodes the total number of *csfs* is $\frac{n \cdot (k-1)}{2}$. Adding the empty forest gives $\frac{n \cdot (k-1)}{2} + 1$. Assume there exist non-empty sequences of sibling nodes S and S' such that $|S| + |S'| = k$. Since $\sum_{i=1}^{|S|} i + \sum_{i=1}^{|S'|} i < \sum_{i=1}^{|S|+|S'|}$, the number of *csfs* is less than $\frac{n \cdot (k-1)}{2}$. This argument holds recursively if there exist more than two sets of siblings. Clearly, if $n \neq p \cdot k$ there exists a sequence of sibling nodes of size less than k and the number of *csfs* is less than $\frac{n \cdot (k-1)}{2} + 1$. Thus, $\frac{n \cdot (k-1)}{2} + 1$ is an upper bound.

Now, I consider the lower bound. If the forest is a tree that has exactly one leaf (there is no branching node) then there are n sequences of sibling nodes, each of length 1. Hence, the number of closed subforests is n. Adding the empty forest results in $n + 1$. Every other tree contains at least one set of siblings with more than one element which increases the number of closed subforests. Thus, $n + 1$ is a lower bound. □

From Theorem 3.1, it follows that Algorithm 1 runs in $O(|F| \cdot \mathit{degree}(F) \cdot |G| \cdot \mathit{degree}(G))$ space. Note that the asymptotic space complexity is not reduced by using dense two-dimensional tables. However, the space reduction can be huge in practice which is measured in Section 4.8.2.

Reducing the Search Space of Forest Alignments

The number of forest alignments grows exponentially with the number of nodes in the tree, which is clear since the forest alignment model is a generalization of the sequence alignment model. In fact, the number of forest alignments exceeds the number of sequence alignments, because a forest alignment is constructed from more combinations of problems. There are two embeddings of a sequence in a forest. In the first, the *vertical embedding*, the parent-child relation of a forest corresponds to the sequence. In the second, *the horizontal embedding*, the sibling order reflects the sequence, i.e. a sequence corresponds to a forest consisting solely of leaves. I now observe how the forest alignment model treats the two embeddings.

Assume a forest F, that begins as a sequence at $F_\perp$ (horizontally or vertically), and some forest G. The general definition of the search space (see Lemma 3.1) considers in Case 2 and Case 3 alignments that differ only in the split position of G whereas both parts are aligned with the empty forest (because F begins as a sequence). In particular, the score of an alignment is build additively from the scores of the edit operations *relabel*, *delete*, *insert*. If $F_\perp$ is a leaf (horizontal embedding), it is unnecessary to align each prefix of G with $F^\downarrow$, the empty forest. Here is why: Nodes of G can be deleted also if the opposing forest is not the empty forest, thus an equivalent alignment of $F^\rightarrow$ and G exists. Accordingly, if F has no right brother (vertical embedding), it is unnecessary to align each suffix of G with $F^\rightarrow$, the empty forest. The following Lemma shows that in these cases the splitting of forests can be omitted and it is still guaranteed to find $\sigma_{\mathrm{FA}}(F, G)$.

Lemma 3.3. *Let A be an alignment of forests F and G. Without loss of generality, assume $label(A_\perp) = (a, \lambda)$. For all $r \in [0, len(G)]$ the following is true:*

- *if $F_\perp$ has no right brother (vertical embedding):*

$$\sigma_{FA}(F^\downarrow, G) \geq \sigma_{FA}(F^\downarrow, r{:}G) + \sigma_{FA}(F^\rightarrow, G{:}(len(G) - r)) \qquad (3.8)$$

- *if $F_\perp$ has no child (horizontal embedding):*

$$\sigma_{FA}(F^{\rightarrow}, G) \geq \sigma_{FA}(F^{\downarrow}, r{:}G) + \sigma_{FA}(F^{\rightarrow}, G{:}(len(G) - r)) \qquad (3.9)$$

Proof. In the vertical embedding, $F^{\rightarrow}$ is the empty forest. Consequently, the alignment of $F^{\rightarrow}$ and $G{:}(len(G) - r)$ consists completely of insertions. There is also an alignment of $F^{\downarrow}$ and G where all nodes in $G{:}(len(G) - r)$ are deleted. Thus, an alignment of $F^{\downarrow}$ and G can achieve at least the same score. The inequality for the horizontal embedding follows analogously. □

If G complies to a sequence, Case 3 of Lemma 3.1 is adjusted accordingly. Figure 3.5 shows the improved recurrence relations for the functions *relabel* and *delete*.

The search space reduction does not reduce the asymptotic time complexity. It is always possible to construct a forest with n nodes where a fractional amount of nodes is neither in the horizontal nor in the vertical embedding. For these nodes the regular recurrences (see Figure 3.4) are applied and, thus, the complexity is not improved. However, in Section 4.8.2 I consider RNA secondary structure forests and measure a constant speedup for these kind of forests.

3.4 Local Similarity in Forests

A global alignment between forests can lead to undesired results if the forests that are aligned share a high similarity only in (coherent) parts but not entirely. Whether the global alignment is the right model or not, depends largely on the data that is observed. The comparison of nucleotide and protein sequences motivated some variants of the sequence alignment model that consider local similar regions in sequences. I concentrate on the *local similarity* and a *small-in-large* variant of this problem [184]. For these problems I give corresponding definitions for forests. It turns out that these variants

$$
\begin{aligned}
\mathit{delete}(F,G) &= \sigma(\mathit{label}(F_\perp),\lambda)\\
&\quad+\begin{cases}
\sigma_{\mathrm{FA}}(F^{\downarrow},G) & \text{if } F_\perp \text{ has no child}\\
\sigma_{\mathrm{FA}}(F^{\rightarrow},G) & \text{if } F_\perp \text{ has no right brother}\\
\max\limits_{r\in[0,\mathit{len}(G)]}\left\{\begin{array}{l}\sigma_{\mathrm{FA}}(F^{\downarrow},r{:}G)\\ +\sigma_{\mathrm{FA}}(F^{\rightarrow},G{:}(\mathit{len}(G)-k))\end{array}\right. & \text{otherwise}
\end{cases}\\
\mathit{insert}(F,G) &= \sigma(\lambda,\mathit{label}(G_\perp))\\
&\quad+\begin{cases}
\sigma_{\mathrm{FA}}(F,G^{\downarrow}) & \text{if } G_\perp \text{ has no child}\\
\sigma_{\mathrm{FA}}(F,G^{\rightarrow}) & \text{if } G_\perp \text{ has no right brother}\\
\max\limits_{r\in[0,\mathit{len}(F)]}\left\{\begin{array}{l}\sigma_{\mathrm{FA}}(r{:}F,G^{\downarrow})\\ +\sigma_{\mathrm{FA}}(F{:}(\mathit{len}(F)-k),G^{\rightarrow})\end{array}\right. & \text{otherwise}
\end{cases}
\end{aligned}
$$

(a)

$$
\begin{aligned}
\mathit{delete}(x,y) &= \sigma(\mathit{lb}_F[i],\lambda)\\
&\quad+\begin{cases}
S_\sigma(\beta_F(i+1,\mathit{noc}_F[i]),\beta_G(k,l)) & \text{if } \mathit{rb}_F[i]=0\\
S_\sigma(\beta_F(\mathit{rb}_F[i],j-1),\beta_G(k,l) & \text{if } \mathit{noc}_F[i]=0\\
\max\limits_{0\le r\le l}\left\{\begin{array}{l}S_\sigma(\beta_F(i+1,\mathit{noc}_F[i]),\beta_G(k,r))\\ +S_\sigma(\beta_F(\mathit{rb}_F[i],j-1),\beta_G(\mathit{rb}^r_G[k],l-r))\end{array}\right. & \text{otherwise}
\end{cases}\\
\mathit{insert}(x,y) &= \sigma(\lambda,\mathit{lb}_G[k])\\
&\quad+\begin{cases}
S_\sigma(\beta_F(i,j),\beta_G(k+1,\mathit{noc}_G[k])) & \text{if } \mathit{rb}_G[k]=0\\
S_\sigma(\beta_F(i,j),\beta_G(\mathit{rb}_G[k],l-1) & \text{if } \mathit{noc}_G[k]=0\\
\max\limits_{0\le r\le j}\left\{\begin{array}{l}S_\sigma(\beta_F(i,r),\beta_G(k+1,\mathit{noc}_G[k]))\\ +S_\sigma(\beta_F(\mathit{rb}^r_F[i],j-r),\beta_G(\mathit{rb}_G[k],l-1))\end{array}\right. & \text{otherwise}
\end{cases}\\
&\text{where } (i,j)=\beta_F^{-1}(x) \text{ and } (k,l)=\beta_G^{-1}(y)
\end{aligned}
$$

(b)

Figure 3.5: *(a) shows improved recurrence relations for the functions delete and insert in search space notation. (b) shows the matrix recurrences. The tables rb_F, noc_F, rb_G and noc_G can be utilized to check whether a node has a child or bother node or not.*

have interesting applications in the comparison of RNA structures (see Section 4.6 and Section 7.1).

Local similarity means finding the maximal similarity between two substructures. If these substructures are extended, the score decreases. This requires a scoring scheme that balances positive and negative scoring contributions. Otherwise, the similarity of the complete structures would always achieve the maximum score. It is generally assumed that an alignment of two empty structures scores zero. A localized variant of distance makes no sense, as empty forests have always the lowest possible distance of zero. A similar problem studied for distance based alignments is the *similar consensus problem* (see Section 2.5.4).

The question is, what are the substructures in a forest? A substring of a string is a prefix of a suffix, and local similarity on strings means the highest similarity over all pairs of substrings. The problem of finding most similar (complete) suffixes is not of great interest in the domain of strings. Moving from strings to forests, local similarity problems come in a greater variety: On trees, the counterpart of a suffix is a subtree. Finding the most similar subtrees *is* an interesting problem, and for forests it generalizes to the problem of finding the most similar closed subforests. Continuing the analogy, the prefix of a (sub)tree T is a tree T' that is obtained by removing subtrees from T which is a tree pattern. I do not consider local similarity of tree patterns in this thesis. However, a pattern similarity algorithms can be *derived* by search space considerations analogous to the presented ones.

3.4.1 Local Closed Subforest Similarity

The *local closed subforest similarity problem* consists in finding the most similar *csfs* F' and G' of F and G. That is:

$$\sigma_{\mathrm{CSF}}(F,G) = \max\{\sigma_{\mathrm{FA}}(F',G') \mid F' \text{ is a } \mathit{csf} \text{ of } F \text{ and } G' \text{ is a } \mathit{csf} \text{ of } G\}. \tag{3.10}$$

For the calculation of global similarity, Algorithm 3.1 calculates the global similarity between all relevant pairs of closed subforests due to Lemma 3.2. This algorithm can be easily modified to calculate the similarity between all combinations of closed subforests by modifying the loop structure.

Iterating over all possible combinations of closed subforests as in Algorithm 3.2 is consistent with the dependencies of the recurrences in Figure 3.4 and Figure 3.5.

```
Input: Forests $F$ and $G$, given by tables
       $lb_F, lb_G, noc_F, noc_G, rb_F, rb_G, offset_F, offset_G,$
       $maxcsflen_F, maxcsflen_G$
Output: $\sigma_{\text{CSF}}(F,G)$ is the maximum value stored in $S_\sigma$
$S_\sigma(0,0) \leftarrow 0$
for $i \leftarrow |F|$ to 1 do
    for $j \leftarrow 1$ to $maxcsflen_F[i]$ do  Calculate $S_\sigma(\beta_F(i,j),0)$
end
for $k \leftarrow |G|$ to 1 do
    for $l \leftarrow 1$ to $maxcsflen_G[k]$ do  Calculate $S_\sigma(0,\beta_G(k,l))$
end
for $i \leftarrow |F|$ to 1 do
    for $k \leftarrow |G|$ to 1 do
        for $j \leftarrow 1$ to $maxcsflen_F[i]$ do
            for $l \leftarrow 1$ to $maxcsflen_G[k]$ do
                Calculate $S_\sigma(\beta_F(i,j),\beta_G(k,l))$
            end
        end
    end
end
```

Algorithm 3.2: Algorithm for the calculation of local closed subforest alignment similarity σ_{CSF}.

Algorithm 3.2 tabulates $\sigma_{\text{FA}}(F',G')$ for a all pairs of *csfs* F' and G' of F and G. Thus, scanning the matrix S_σ for maximum elements yields to $\sigma_{\text{CSF}}(F,G)$. An optimal alignment can be obtained by backtracking.

Space and Time Complexity I apply the same tabulation technique as for the calculation of global similarity. Hence, the space complexity is the same as for Algorithm 3.1, $O(|F| \cdot \mathit{degree}(F) \cdot |G| \cdot \mathit{degree}(G))$. However, now each element of S_σ is calculated and the tabulation is dense.

According to the recurrences in Figure 3.4, each element of S_σ is calculated in $O(\mathit{degree}(F) + \mathit{degree}(G))$ time. Since each entry in S_σ is calculated exactly once, the overall time complexity of Algorithm 3.2 depends on the size of S_σ. This in turn depends on the number of *csfs* in F and G which was analyzed in Theorem 3.1. Consequently, Algorithm 3.2 runs in $O(|F| \cdot |G| \cdot \mathit{degree}(F) \cdot \mathit{degree}(G) \cdot (\mathit{degree}(F) + \mathit{degree}(G)))$ time.

3.4.2 Small-in-Large Closed Subforest Similarity

The *small-in-large closed subforest similarity* is the maximum similarity between a (smaller) forest F and a *csf* G' of G. That is,

$$\sigma_{\mathrm{SIL\_CSF}}(F, G) = \max\{\sigma_{\mathrm{FA}}(F, G') \mid G' \text{ is a } \mathit{csf} \text{ of } G\}. \tag{3.11}$$

Algorithm 3.2 calculates the similarity between all closed subforests and, since F is also a *csf*, also between F and all *csfs* of G. Thus, scanning the matrix S_σ for maximum elements $S_\sigma(x, y)$, such that $x = \beta_F(\alpha_F(F))$, yields to $\sigma_{\mathrm{CSF}}(F, G)$.

Space and Time Complexity The space and time complexity is the same as for the calculation of σ_{CSF} since the scanning of matrix S_σ is the only difference. The time complexity could be further reduced since, in analogy to the global similarity, not each pair of closed subforests is a relevant subforest. A combination of Algorithm 3.1 and Algorithm 3.2 that has a loop structure as in Algorithm 3.1 for forest F and as in Algorithm 3.2 for forest G would reduce the time complexity. I do not provide a formal analysis since the time improvement relies on the smaller of both forests and Algorithm 3.2 yields to good practical runtime (see Section 4.8.2).

3.4.3 Suboptimal Solutions

Possibly, there is more than one highly similar region in two forests or that a smaller forest appears more than once in a larger one. If so, all solutions above a certain threshold of similarity are of interest. To avoid redundant alignments, i.e. alignments that "intersect" with previously reported solutions are excluded. I define *csfs* F' and F'' of F to *intersect* if they share a common node. For the calculation of small-in-large similarity, the non-intersecting property is obsolete for the smaller forest F. The test for intersection can be easily done in the index pair representation of *csfs*.

The generation of suboptimal solutions within a percentage t of the optimal solution follows a simple procedure: For each reported solution the local similar *csfs* of F and G are stored in lists L_F and L_G, respectively. A solution is reported if it is within t percent of the optimal score and at least one of the local similar forests does not intersect with forest in L_F or L_G. The solutions are considered in order of their closeness to the optimum, starting with the optimum.

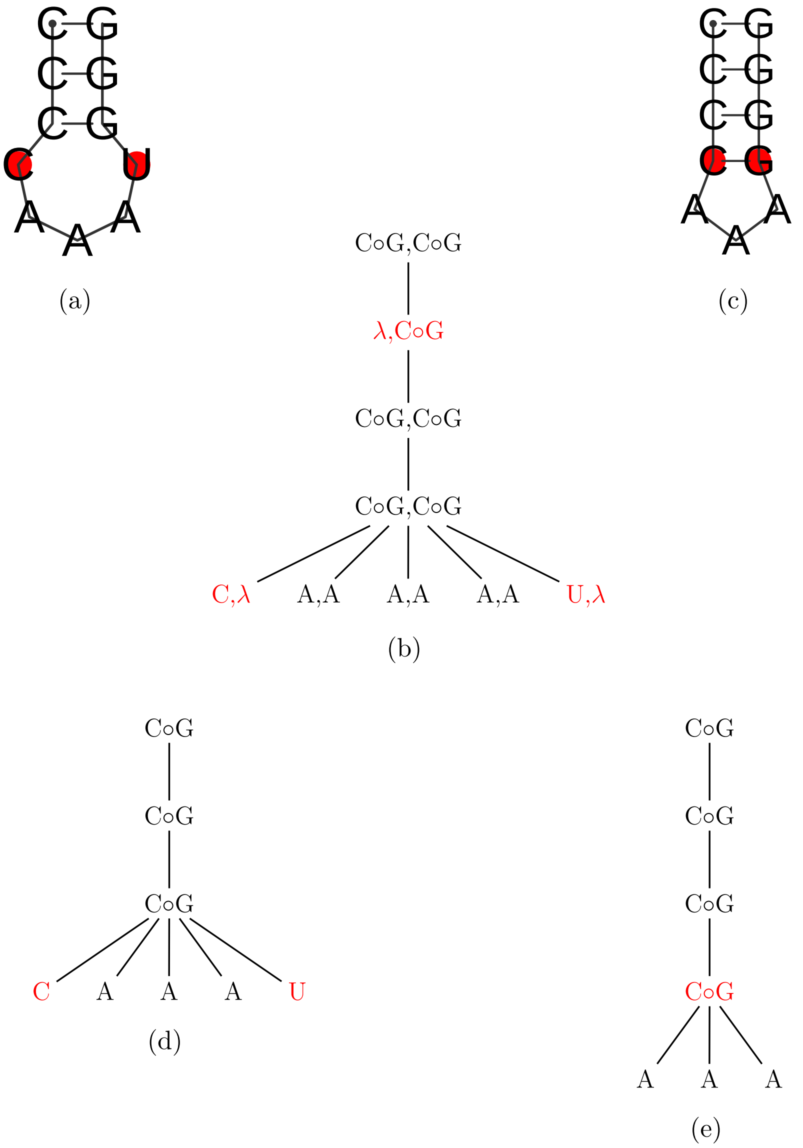
Chapter 4

Pairwise Comparison of RNA Secondary Structures in the Forest Alignment Model

The tree alignment distance can be applied straightforward to the natural and coarse tree representation, resulting in a distance measure on RNA secondary structures. The coarse grained representation produces smaller trees than the natural tree representation, which in turn reduces the practical runtime. On the other hand, the natural tree representation allows for simpler scoring functions, i.e. edit operations on structural components of an RNA are hard to motivate biologically since these are not the entities on which the evolution of a structure actually happens. Mutations happen on the nucleotide level and bases and base-pairings are subject to the selective pressure. Beside a structural alignment, an alignment of trees in the natural representation produces an alignment of nucleotide sequences. Such an alignment is desirable to analyze the evolution of a structural RNA which happens on the sequence level. For example, compensatorial mutations in stem regions or sequence variations in loop regions can be detected. However, a base-pair breaking, i.e. a bond between two bases becomes too weak (refer to Section 2.5.5), cannot be modeled adequately by aligning RNA structures in the natural tree representation. Here is why: A base-pair is represented as a single node and two unpaired bases are represented as two single nodes. In a tree alignment, each node of the aligned trees is involved in exactly one edit operation. It can be relabeled, deleted or inserted. Thus, it is not possible to associate a base-pair with two unpaired bases. Figure 4.1 gives a concrete example.

These limitations of tree based approaches to compare RNA secondary structures gave actually rise to the arc-annotated sequence approaches (see Section 2.5.5). Here, I introduce a new forest representation[1] for RNA secondary structures that, in conjunction with a slight modification of the forest alignment model, allows for explicit scoring of base-pair breakings. Beside a global RNA structure alignment, I introduce local variants in analogy to Section 3.4. I demonstrate the performance of my algorithms by providing exhaustive measurements concerning the practical runtime and memory consumption. An intuitive 2d-plot for RNA secondary structure alignments makes the results of a structural comparison usable without requiring knowledge in abstract structure representations. A prediction strategy for structural motifs in RNA molecules that, at its heart, uses local structure alignments is provided in Section 7.1.

[1] A tree representation can be defined analogously. For convenience, I switch to forests again.



(a)

(c)

(b)

(d)

(e)

Figure 4.1: *The RNA structures (a) and (c) differ in the length of the stacking region and the loop size. A reasonable theory for their evolution is that the red U in (a) mutated to G in (c), allowing for an additional base-pair. (d) and (e) show the tree representation of (a) and (c), respectively. (b) shows an alignment of these trees. Each node of a tree is involved in exactly one edit operation in a tree alignment. Since a base-pair is encoded as a single node, the score for deleting the pairing between bases a and b is* $\sigma(a{\circ}b, \lambda) + \sigma(\lambda, a) + \sigma(\lambda, b)$*. The insertions and deletions are not "coordinated" in (b), i.e. the base-pair G∘C is inserted some bases away from the deleted free bases. Another consequence is that sequence similarity cannot contribute to an optimal alignment in a base-pair breaking operation since bases are never relabeled but inserted and deleted.*

4.1 Preliminaries

I extend the notation introduced in Section 3.1: I define $F_{\overset{\rightarrow}{\bot}}$ to be the root node of the rightmost tree in a forest F. The forest $\circ\!|F|\!\circ$ denotes the forest F omitting the leftmost and the rightmost tree.

4.2 The Extended Forest Representation of RNA Secondary Structures

The *extended forest representation* extends the natural tree representation of RNA secondary structures[2] such that base-pairs are represented by three connected nodes: The *pair-node*, for short *P-node*, stands for a base-pair bond and is labeled with P. Its children nodes are ordered according to the $5'$ to $3'$ ordering of bases and the leftmost and the rightmost child are the bases that pair. A node that is not a P-node is a *base-node*, for short *B-node*, and is labeled with one of the bases A,C,G,U. Note that the children nodes of a P-node can be P-nodes except for the leftmost and the rightmost child. Hence, a P-node is always an internal node, whereas a B-node is always a leaf. In this sense, the structure given by the P-nodes is imposed on the sequence of B-nodes. Figure 4.2 gives an example of the extended RNA forest representation which is in flavor of parse trees for context free grammars that describe RNA secondary structures [37, 168].

[2] Precisely, the forest counterpart.

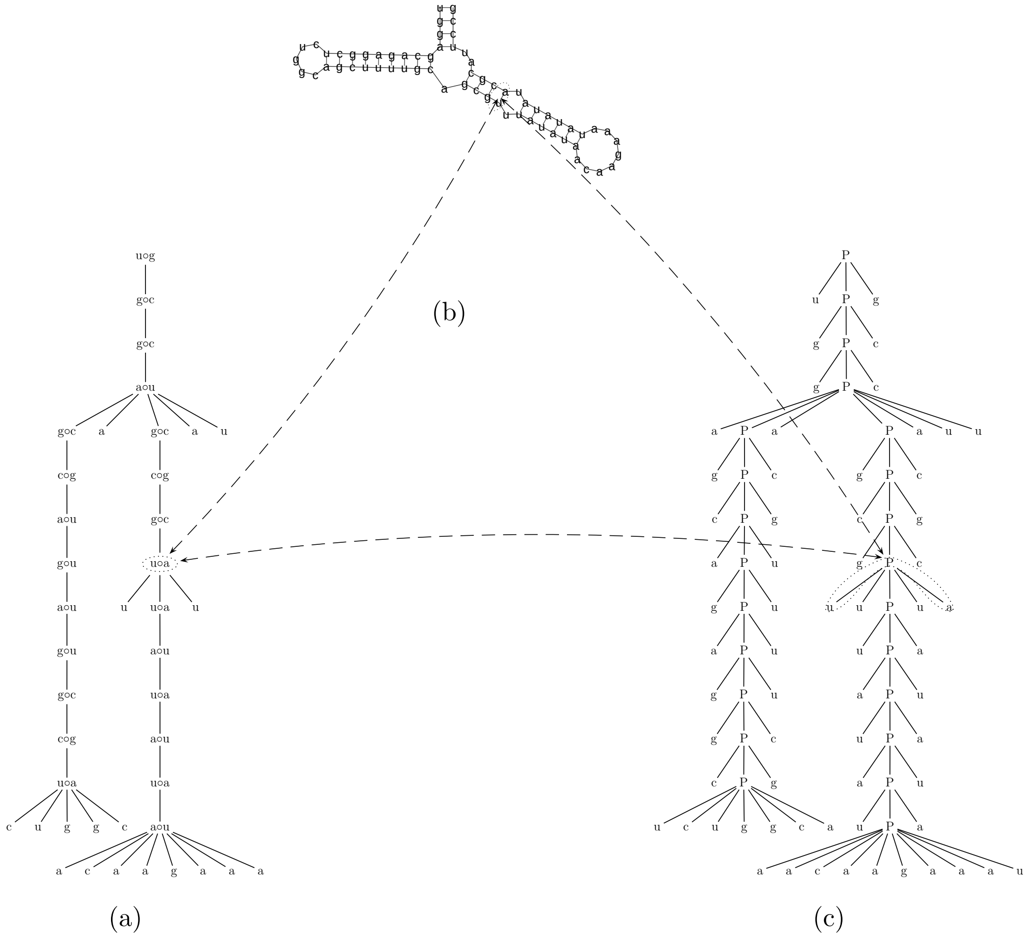


Figure 4.2: *(a) shows the natural tree representation of structure (b), and (c) shows the extended forest representation of structure (b) (in this case a tree).*

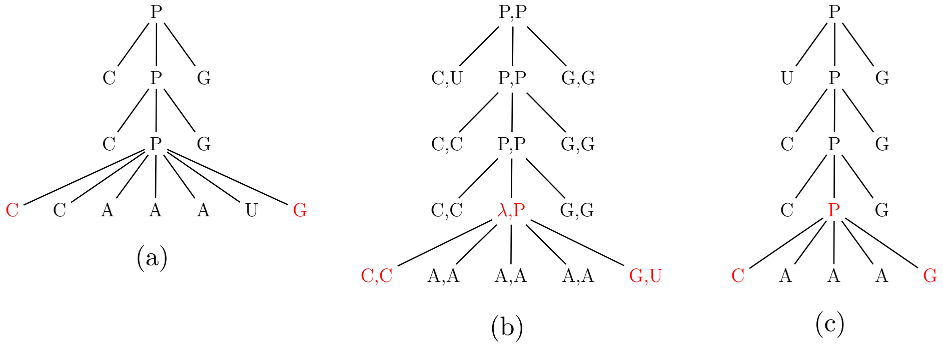


Figure 4.3: *(a) and (c) show the extended forest representations of the structures shown in Figure 4.1(a) and 4.1(c), respectively. The alignment (b), which is optimal for a scoring scheme that favors base matches, accounts for the base-pair breaking by the deletion of a P-node, a match of the conserved base, and a mismatch of the mutated base. Thus, the scoring contribution is* $\sigma(\lambda, P) + \sigma(C, C) + \sigma(G, U)$.

4.3 The Welformed Alignment Model

An alignment of forests in the extended forest representation is a simultaneous alignment of sequence and structure. The sequence and structure alignment mutually improves the quality of the whole sequence-structure alignment. Figure 4.3 shows how a forest alignment using the extended forest representation can account for the differences of the structures studied in Figure 4.1.

However, a straightforward application of the classical forest alignment model (see Section 3.2) to the extended forest representation of RNA secondary structures results in the following problems:

1. A match of a P-node with a B-node cannot be interpreted as an edit operation on RNA structures.

2. It is not guaranteed by the model that, once a P-node is matched, the corresponding paired bases are relabeled by each other. See Figure 4.4 for an example of an alignment that should be avoided.

3. The score of a base-pair replacement is built from the independent scores of a P-node match and the matches (or replacements) of the

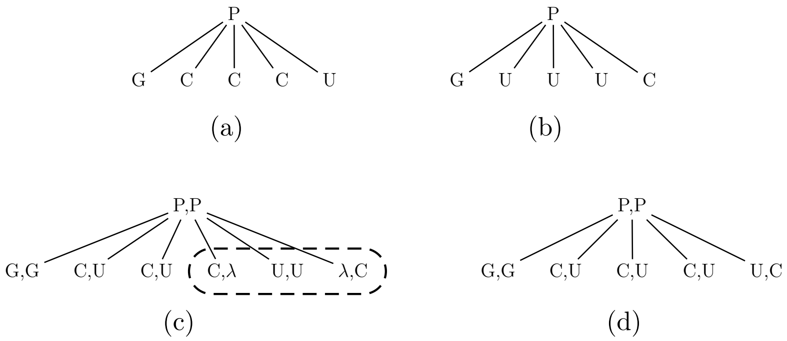


Figure 4.4: *Consider a feasible scoring scheme that gives better scores to base matches than to mismatches and indels. (c) shows an alignment of (a) and (b) that is not welformed, i.e. the base-pairs $G \circ U$ and $G \circ C$ are not aligned on the sequence level, though the corresponding P-nodes are matched. (d) shows a welformed tree alignment of the same structures where paired bases are aligned to each other.*

paired bases. Thus, an empirical derived scoring scheme based on base-pair substitution frequencies as proposed by Klein & Eddy cannot be used [103].

Case 1 can be easily avoided by assigning an infinite negative score to a relabeling of a P-node with a B-node. The limitations explained in Case 2 and Case 3 result from the following fact: A base-pair replacement is not an elementary edit operation in the tree edit model but affects three nodes in the extended forest representation. In analogy to the base-pair replace operation in the general edit model for RNA secondary structures (see Section 2.5.5), I introduce a base-pair replace operation for the extended forest representation. I extend Tai's edit model (see Section 2.5.3) introducing a new edit operation for the P-nodes in the extended forest representation. Intuitively, this means whenever a P-node is matched, the corresponding paired bases are relabeled by each other. I refine the *relabel* function by introducing two new edit operations that can be applied to either P-nodes or B-nodes:

- *basepair relabel*: Two P-nodes are matched and their leftmost and rightmost children are relabeled by each other.
- *base relabel*: The label of a B-node is replaced by another, possibly the

same, B-node label.

An alignment that results from this extended edit model is denoted a *welformed alignment*. It is obvious that the following holds:

$$\text{An alignment } A \text{ of forests } F \text{ and } G \text{ is a } \textit{welformed forest alignment} \text{ iff}$$
$$\text{for all relabeled P-nodes in } A \text{ the corresponding B-nodes are relabeled in } A. \tag{4.1}$$

The *welformed tree alignment similarity* σ_{WFA} is defined as:

$$\sigma_{\text{WFA}}(F,G) = \max\{\sigma(A) \mid \text{A is a welformed forest alignment of } F \text{ and } G\}. \tag{4.2}$$

4.4 A Global RNA Secondary Structure Alignment Algorithm

4.4.1 The Search Space of RNA Secondary Structure Alignments

Remember the observations concerning the search space of forest alignments in Lemma 3.1. Clearly, the search space definitions of Case 2 (delete) and Case 3 (insert) are not affected by the constraints that make an alignment a welformed alignment. Case 1 treats the relabeling of nodes and I refine this case to be consistent with the definition of welformed forest alignments. The following case analysis replaces Case 1 in Lemma 3.1 resulting in a definition of the search space for welformed forest alignments.

Lemma 4.1. *Let A be a welformed alignment of forests F and G in the extended forest representation such that $(a,b) = label(A_\perp)$ and $a, b \notin \lambda$. Let $(c,d) = A^{\downarrow}{}_{\perp}$ and $(e,f) = A^{\downarrow}{}_{\vec{\perp}}$.*

1. *if $label(A_\perp) = (P,P)$ then the following is true:*

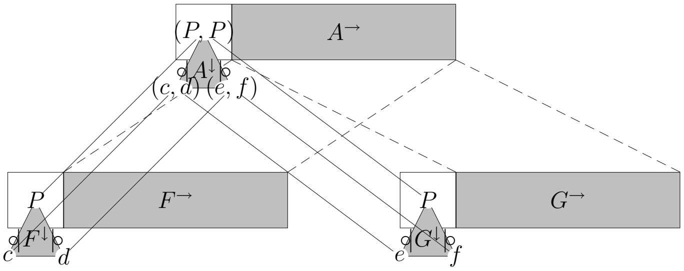

(P, P)
$A^{\rightarrow}$
$\circ\!|A^{\downarrow}|\!\circ$
(c, d) (e, f)
P
$F^{\rightarrow}$
$\circ\!|F^{\downarrow}|\!\circ$
c
d
P
$G^{\rightarrow}$
$\circ\!|G^{\downarrow}|\!\circ$
e
f

Figure 4.5: *Search space implications of Lemma 4.1 are indicated by the lines connecting alignment A and forests F and G. The lines connecting $\circ\!|A^{\downarrow}|\!\circ$, $\circ\!|F^{\downarrow}|\!\circ$ and $\circ\!|G^{\downarrow}|\!\circ$ are omitted.*

- *label*$(F_{\perp}) = P$ *and label*$(G_{\perp}) = P$*,*
- $c = F^{\downarrow}{}_{\perp}, d = F^{\downarrow}{}_{\vec{\perp}}, e = G^{\downarrow}{}_{\perp}$ *and* $f = G^{\downarrow}{}_{\vec{\perp}}$*,*
- $\circ\!|A^{\downarrow}|\!\circ$ *is an alignment of* $\circ\!|F^{\downarrow}|\!\circ$ *and* $\circ\!|G^{\downarrow}|\!\circ$*.*

2. *if label*$(A_{\perp}) \neq (P, P)$ *then* $A^{\rightarrow}$ *is an alignment of* $F^{\rightarrow}$ *and* $G^{\rightarrow}$*.*

Proof. Case 1 follows directly from the definition of welformed alignments. Case 2 handles the relabeling of B-nodes and is a special case of Case 1 in Lemma 3.1. In this case, $F_{\perp}$ and $G_{\perp}$ are leaves and an alignment of the children forest is always the empty forest. □

Figure 4.5 illustrates the search space implications that follows from Lemma 4.1. The recursive function for the new relabel operation follows directly from Lemma 4.1 and is shown in Figure 4.6 (a).

4.4.2 Implementation based on Matrix Recurrences

The calculation of matrix elements as in Algorithm 3.1 is not directly suitable for the calculation of welformed forest alignment similarity. The reason is that the calculation of the similarities between all pairs of relevant *csfs* due to Lemma 3.2 is not sufficient for the recurrences in Figure 4.6 (a). In explanation, the recurrences in Figure 4.6 (a) can lead to pairs of *csfs* $(\circ\!|\bar{F}|\!\circ, \circ\!|\bar{G}|\!\circ)$ which are not relevant due to Lemma 3.2 and, hence, are not calculated by

Algorithm 3.1. The following Lemma identifies the relevant subforest for the calculation of $\sigma_{\mathrm{WFA}}(F,G)$.

Lemma 4.2. *The pairs of subforests that are relevant for the calculation of $\sigma_{WFA}(F,G)$ due to the combined recurrences in the Figures 3.2 and 4.6 have the form $(\bar{F}{:}j,\, {}_k]\bar{G}[_l)$, $({}_i]\bar{F}[_j,\, \bar{G}{:}l)$, $({\circ\!|}\bar{F}{|\!\circ}{:}j,\, {}_k]\bar{G}[_l)$, and $({}_i]\bar{F}[_j,\, {\circ\!|}\bar{G}{|\!\circ}{:}l)$ where $\bar{F}$ and $\bar{G}$ are maximal closed subforests of F and G, respectively.*

Proof. I consider all possible transitions to subforests resulting from the definition of the search space of welformed forest alignments. These are:

- Case 1 (from Lemma 4.1): $({\circ\!|}F^{\downarrow}{|\!\circ}, {\circ\!|}G^{\downarrow}{|\!\circ}), (F^{\rightarrow}, G^{\rightarrow})$
- Case 2 (from Lemma 3.1): $(F^{\downarrow}, k{:}G), (F^{\rightarrow}, G{:}l)$
- Case 3 (from Lemma 3.1): $(i{:}F, G^{\downarrow}), (F{:}j, G^{\rightarrow})$

Referring to the transition graph in the proof of Lemma 3.2, all pairs of subforests $(\bar{F}{:}j,\, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{i}}]\bar{F}[_{\mathrm{j}},\, \bar{G}{:}l)$ can be reached by transitions to subforests, even though there is no $(F^{\downarrow}, G^{\downarrow})$ transition. The transition $({\circ\!|}F^{\downarrow}{|\!\circ}, {\circ\!|}G^{\downarrow}{|\!\circ})$ results in pairs of forests of the form $({\circ\!|}\bar{F}{|\!\circ}, {\circ\!|}\bar{G}{|\!\circ})$. Considering the possible transition in Lemma 3.2 again, this can lead to the pairs of subforest $({\circ\!|}\bar{F}{|\!\circ}{:}j,\, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{i}}]\bar{F}[_{\mathrm{j}},\, {\circ\!|}\bar{G}{|\!\circ}{:}l)$. Hence, Lemma 4.2 identifies the relevant pairs of subforests for the calculation of $\sigma_{\mathrm{WFA}}(F,G)$. □

Obviously, each relevant subforest for the calculation of $\sigma_{\mathrm{WFA}}(F,G)$ is a *csf* and the tabulation technique introduced in Section 3.3.2 can be applied to derive a dynamic programming algorithm. Figure 4.6 (b) shows the matrix recurrences for the relabel operation in the welformed forest alignment model. For constant time access to the rightmost child of a P-node, the new table rmb_F stores at position i the preorder index of the rightmost brother node. Note that rmb_F is only looked up for the leftmost child of a P-node. This is why a one dimensional lookup table is sufficient.

$$relabel(F,G) = \begin{cases} \begin{array}{l} \sigma(label(F_{\perp}), label(G_{\perp})) \\ +\sigma(label({F^{\downarrow}}_{\perp}), label({G^{\downarrow}}_{\perp})) \\ +\sigma(label({F^{\downarrow}}_{\overset{\rightarrow}{\perp}}), label({G^{\downarrow}}_{\overset{\rightarrow}{\perp}})) \\ +\sigma_{\mathrm{WFA}}(\circ\!|F^{\downarrow}|\!\circ, \circ\!|G^{\downarrow}|\!\circ) \\ +\sigma_{\mathrm{WFA}}(F^{\rightarrow}, G^{\rightarrow}) \end{array} & \text{if } label(F_{\perp}) = P \text{ and } label(G_{\perp}) = P \\ \\ \begin{array}{l} \sigma(label(F_{\perp}), label(G_{\perp})) \\ +\sigma_{\mathrm{WFA}}(F^{\rightarrow}, G^{\rightarrow}) \end{array} & \text{otherwise} \end{cases}$$

(a)

$$relabel(x,y) = \begin{cases} \begin{array}{l} \sigma(lb_F[i], lb_G[k]) \\ +\sigma(lb_F[i+1], lb_G[k+1]) \\ +\sigma(lb_F[rmb_F[i]], lb_F[rmb_G[k]]) \\ +S_{\sigma}(\beta_F(rb_F[i+1], noc_F[i]-2), \beta_G(rb_G[k+1], noc_G[k]-2)) \\ +S_{\sigma}(\beta_F(rb_F[i], j-1), \beta_G(rb_G[k], l-1)) \end{array} & \text{if } lb_F[i] = P \text{ and } lb_G[k] = P \\ \\ \begin{array}{l} \sigma(lb_F[i], lb_G[k]) \\ +S_{\sigma}(\beta_F(rb_F[i], j-1), \beta_G(rb_G[k], l-1)) \end{array} & \text{otherwise} \end{cases}$$

$$\text{where } (i,j) = \beta_F^{-1}(x) \text{ and } (k,l) = \beta_G^{-1}(y)$$

(b)

Figure 4.6: *Recurrences for the refined relabel function for welformed forest alignments resulting from the search space observations in Lemma 4.1. (a) shows the recursive formula in search space notation. (b) shows the corresponding matrix recurrences using the tabulation technique introduced in Section 3.3.2.*

The Order of Calculation

A straightforward solution would be to use Algorithm 3.2 with the refined relabel recurrences in Figure 4.5 (b). However, for the calculation of global similarity this would increase the time complexity unnecessarily since Algorithm 3.2 calculates the similarity between all pairs of closed subforests. Algorithm 4.1 calculates only the closed subforests that are relevant for the calculation of the global welformed forest alignment similarity. This algorithm is obtained by including the missing calculations for the welformed alignment model in Algorithm 3.1.

Efficiency Analysis

The space and time efficiency for the calculation of $\sigma_{\text{WFA}}(F, G)$ is the same as for the corresponding version of the classical forest alignment similarity (see Section 3.3.2). This is obvious since the refined relabel operation can be calculated in constant time, the loop structure of Algorithm 3.1 and Algorithm 4.1 is the same, and the same tabulation technique is used. Restating the complexity results, the time complexity for the calculation of $\sigma_{\text{WFA}}(F, G)$ is $O(|F| \cdot |G| \cdot (\mathit{degree}(F) + \mathit{degree}(G))^2)$ and the space complexity is $O(|F| \cdot \mathit{degree}(F) \cdot |G| \cdot \mathit{degree}(G))$. In Section 4.8.2, I will observe how the critical parameters, size and degree, scale for the extended forest representation of RNA secondary structures.

4.5 Scoring Schemes

The extended forest representation is suitable to score both structure and sequence similarity. The structural edit operations affect the P-nodes and sequence edit operations affect the B-nodes. I will present two scoring schemes: First, a pure structure based scoring where sequence information is neglected or only contributes marginally to the overall score, such that structure is dominating. The scoring contributions of a base-pair is built from the independent scores of the aligned P-node and B-nodes. Second, I employ a

```
   Input: Forests $F$ and $G$, stored as tables
          $lb_F, lb_G, noc_F, noc_G, rb_F, rb_G, offset_F, offset_G,$
          $maxcsflen_F, maxcsflen_G, rmb_F, rmb_G$
   Output: $\sigma_{\mathrm{WFA}}(F,G)$ stored at
          $S_\sigma(\beta_F(i, maxcsflen_F[i]), \beta_G(k, maxcsflen_G[k]))$
 1 $S_\sigma(0,0) \leftarrow 0$
 2 for $i \leftarrow |F|$ to 1 do
 3     for $j \leftarrow 1$ to $maxcsflen_F[i]$ do  Calculate $S_\sigma(\beta_F(i,j), 0)$
 4 end
 5 for $k \leftarrow |G|$ to 1 do
 6     for $l \leftarrow 1$ to $maxcsflen_G[k]$ do  Calculate $S_\sigma(0, \beta_G(k,l))$
 7 end
 8 for $i \leftarrow |F|$ to 1 do
 9     for $k \leftarrow |G|$ to 1 do
10         for $j \leftarrow 1$ to $maxcsflen_F[i]$ do
11             Calculate $S_\sigma(\beta_F(i,j), \beta_G(rb_G[k], maxcsflen_G[k]-1))$
12             Calculate $S_\sigma(\beta_F(i,j), \beta_G(k, maxcsflen_G[k]))$
13         end
14         for $l \leftarrow 1$ to $maxcsflen_G[k]$ do
15             Calculate $S_\sigma(\beta_F(rb_F[i], maxcsflen_F[i]-1), \beta_G(k,l))$
16             Calculate $S_\sigma(\beta_F(i, maxcsflen_F[i]), \beta_G(k,l))$
17         end
18     end
19 end
```

Algorithm 4.1: Algorithm for the calculation of welformed global forest alignment similarity σ_{WFA}. The calculation of elements of S_σ includes the recurrences in Figure 4.6 (b). In comparison to Algorithm 3.1, there are additional computations in Line 11 and Line 15 for the similarity calculations of the pairs $(\circ\!|\bar{F}|\!\circ : j, {}_{\mathrm{k}}]\bar{G}[_{\mathrm{l}})$ and $({}_{\mathrm{i}}]\bar{F}[_{\mathrm{j}}, \circ\!|\bar{G}|\!\circ : l)$. Note that if, e.g. in Line 11, k has no right brother, the term $\beta_G(rb_G[k], maxcsflen_G[k]-1)$ evaluates to $\beta_G(0,0)$ which is the empty forest. This calculation is not necessary since the alignments involving the empty forest are already calculated in Line 1-7. However, the alternative would be a case distinction in the loop structure which makes the algorithm more complicated.

$$
\begin{array}{lllllll}
 & \mathrm{A} & \mathrm{C} & \mathrm{G} & \mathrm{U} & \mathrm{P} & \lambda \\
\mathrm{A} & 1 & & & & & \\
\mathrm{C} & 0 & 1 & & & & \\
\mathrm{G} & 0 & 0 & 1 & & & \\
\mathrm{U} & 0 & 0 & 0 & 1 & & \\
\mathrm{P} & -\infty & -\infty & -\infty & -\infty & 10 & \\
\lambda & \text{-10} & \text{-10} & \text{-10} & \text{-10} & \text{-5} & n.d.
\end{array}
$$

Figure 4.7: *Scoring values for the scoring function σ_P. Since scoring functions are generally considered to be symmetric, a triangle matrix is sufficient to define the scoring function. The substitution of λ by λ is not defined since it never happens in an alignment model.*

scoring scheme based on empirically derived substitution scores for aligned bases and base–pairs, the *RIBOSUM* score. In this scoring scheme, the aligned bases in a base-pair replacement are considered simultaneously.

4.5.1 Pure Structure Alignment Score

A *pure structure alignment* of RNA secondary structures is an alignment due to a scoring scheme that is guided by the structure and not by the sequence. However, it makes sense that, especially in aligned loop regions, sequence information can be used to improve the results. Therefore, I give a positive score to a base-match which is much smaller than the score for a base-pair match, or precisely, the match of a base-pair bond represented by a P-node. The score of a base-pair replacement is built from the match of two P-nodes plus the replacement scores of the involved bases (refer to the recurrences in Figure 4.6). Clearly, the deletion of base-pair bonds (P-nodes) and the deletion of bases (B-nodes) should be penalized, but the deletion of a base-pair bond should not cost as much as the deletion of a base. Based on these considerations, I define the scoring function σ_P given by the scoring matrix in Figure 4.7.

4.5.2 *RIBOSUM* Scores

The scoring of sequence alignments received much attention and a good scoring scheme is a prerequisite to produce biological meaningful alignments especially for protein sequences. For sequences, log-odds position independent substitution matrices were successfully applied to compute the alignment scores. Most prominent are *BLOSUM*[3] and *PAM*[4] matrices [31, 75]. The former is generally acknowledged to produce better results for evolutionary distantly related sequences.

Recently, Klein & Eddy generalized Henikoff & Henikoff's *BLOSUM* idea to structural RNA resulting in two substitution matrices: One for unpaired bases and one for base-pairs. According to the *BLOSUM* matrixes, they called their scoring matrices *RIBOSUM* (**RIBO**somal RNA **SU**bstitution **M**atrix) [103]. I now resemble the idea for the calculation of *RIBOSUM* matrices and how they can be used in my structure comparison algorithms.

The substitution scores are empirically derived from hand-crafted high-quality alignments of the small subunit RNA from the *European Ribosomal RNA Database* [32]. The scoring matrices give the log-odds ratio for observing a given substitution relative to background nucleotide frequencies. For single base substitutions this is a 4×4 matrix S given by

$$s_{ij} = \log_2 \frac{f_{ij}}{g_i \cdot g_j}, \tag{4.3}$$

where i and j are the two aligned nucleotides, f_{ij} is the empirically observed frequency of i aligned to j in homologous RNAs, and g_i and g_j are the background frequencies of the individual nucleotides. For base-pair substitutions this is a 16×16 matrix S' given by

$$s'_{ijkl} = \log_2 \frac{f'_{ijkl}}{g_i \cdot \; g_j \cdot g_k \cdot g_l}, \tag{4.4}$$

[3] **BLO**ck **SU**bstitution **M**atrix
[4] **P**ercent **A**ccepted **M**utations

where i is base-paired to j, k is base-paired to l, i is aligned with k, and j is aligned with l. In this case, f'_{ijkl} is the observed frequency of the two base pairs i∘j and k∘l aligned to each other in homologous RNAs. g again is the background frequency of the individual nucleotides.

A naive counting of the frequencies f_{ij} and f'_{ijkl} could bias the substitution scores towards overrepresented sub-families in the alignment. To eliminate this risk, clusters of similar sequences are formed that weight the individual sequences in the alignment, i.e. a member of a large cluster has a small weight. A single linkage clustering technique groups sequences with a percentage identity above some threshold x. To allow shorter evolutionary distances than the original *BLOSUM* matrices, Klein & Eddy added a second sequence identity cutoff y. Only pairs of sequences that exceed y percent identity are counted at all. By adjusting x and y, a specific *RIBOSUMx-y* matrix can be constructed. Klein & Eddy observed that the *RIBSOUM85-60* matrix (see Figure 4.8) is a good *ab initio* choice.

The recurrences in Figure 4.6 can be easily adapted to score base-pair substitutions. Instead of adding the scores for a P node and the aligned bases, a scoring function σ_{BP} accepts the whole base-pairs as its parameter. In particular in Figure 4.6 (a) the terms

$$\sigma(label(F_{\perp}), label(G_{\perp})) + \sigma(label(F^{\downarrow}{}_{\perp}), label(G^{\downarrow}{}_{\perp})) + \sigma(label(F^{\downarrow}{}_{\vec{\perp}}), label(G^{\downarrow}{}_{\vec{\perp}}))$$

are substituted by

$$\sigma_{BP}(label(F^{\downarrow}{}_{\perp}) \circ label(F^{\downarrow}{}_{\vec{\perp}}), label(G^{\downarrow}{}_{\perp}) \circ label(G^{\downarrow}{}_{\vec{\perp}})).$$

The corresponding holds for the table based recurrences in Figure 4.6 (b). A problem that remains is to set the score for P-node and B-node deletions. I set both to -2. However, these parameters should be empirically adjusted for the particular application.

	A	C	G	U
A	2.22			
C	-1.86	1.16		
G	-1.46	-2.48	1.03	
U	-1.39	-1.05	-1.74	1.65

(a)

	AA	AC	AG	AU	CA	CC	CG	CU	GA	GC	GG	GU	UA	UC	UG	UU
AA	-2.49															
AC	-7.04	-2.11														
AG	-8.24	-8.89	-0.80													
AU	-4.32	-2.04	-5.13	**4.49**												
CA	-8.84	-9.37	-10.41	-5.56	-5.13											
CC	-14.37	-9.08	-14.53	-6.71	-10.45	-3.59										
CG	-4.68	-5.86	-4.57	1.67	-3.57	-5.71	**5.36**									
CU	-12.64	-10.45	-10.14	-5.17	-8.49	-5.77	-4.96	-2.28								
GA	-6.86	-9.73	-8.61	-5.33	-7.98	-12.43	-6.00	-7.71	-1.05							
GC	-5.03	-3.81	-5.77	**2.70**	-5.95	-3.70	**2.11**	-5.84	-4.88	**5.62**						
GG	-8.39	-11.05	-5.38	-5.61	-11.36	-12.58	-4.66	-13.69	-8.67	-4.13	-1.98					
GU	-5.84	-4.72	-6.60	**0.59**	-7.93	-7.88	-0.27	-5.61	-6.10	**1.21**	-5.77	**3.47**				
UA	-4.01	-5.33	-5.43	**1.61**	-2.42	-6.88	**2.75**	-4.72	-5.85	**1.60**	-5.75	-0.57	**4.97**			
UC	-11.32	-8.67	-8.87	-4.81	-7.08	-7.40	-4.91	-3.83	-6.63	-4.49	-12.01	-5.30	-2.98	-3.21		
UG	-6.16	-6.93	-5.94	-0.51	-5.63	-8.41	**1.32**	-7.36	-7.55	-0.08	-4.27	-2.09	**1.14**	-4.76	**3.36**	
UU	-9.05	-7.83	-11.07	-2.98	-8.39	-5.41	-3.67	-5.21	-11.54	-3.90	-10.79	-4.45	-3.39	-5.97	-4.28	-0.02

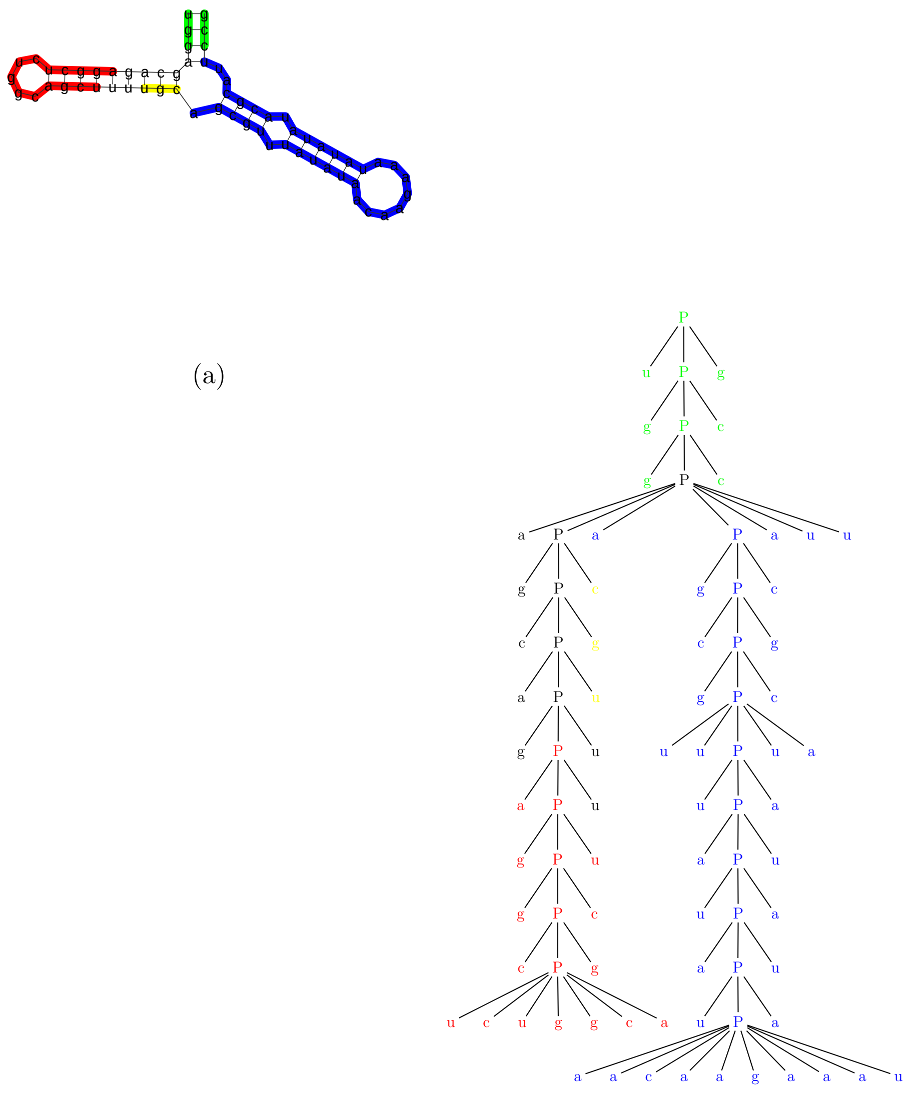
(b)

Figure 4.8: RIBOSUM85-60 *matrix. Watson-Crick base-pairs substitutions are emphasized.*

4.6 Local Similarity in RNA Structures

In Section 3.4, two local similarity algorithms for forests were presented: The closed subforest similarity σ_{CSF} and the small-in-large similarity $\sigma_{\text{SIL\_CSF}}$. Since I represent RNA secondary structures as forests, these local similarity algorithms can be used directly to find local similarities in RNA secondary structures. The local substructures that are considered are closed subforests of the forest representation. Figure 4.9 explains what kind of substructures of RNA secondary structures corresponds to closed subforests.

Algorithm 3.2 calculates both the local similarities σ_{CSF} and $\sigma_{\text{SIL\_CSF}}$ in RNA secondary structures in the extended forest representation. The only difference is that the recurrences for the relabel function in welformed forest alignment (Figure 4.6) replace the relabel function in the classic forest alignment model (Figure 3.4). The control structure of Algorithm 3.2 remains the same and so does the efficiency.



(a)

(b)

Figure 4.9: *Closed subforests correspond to closed substructures in RNA secondary structures. Intuitively, this means that the substructures are contiguous and "closed" by hairpin loops. Closed subforests are sequences of consecutive subtrees. In my definition, a subtree contains all edges and nodes emanating from its root. The blue regions shows a substructure in (a) that corresponds to a closed subforest in (b). The green part of the structure is not a closed subforest because the descending nodes are not included, it is not closed. The red part shows a substructure that is not considered as a local structure for the same reason. However, this is less obvious since only the U, which is a child of the root of this subtree, is not included. If the top-level P node would not be included in the red substructure, this part would correspond to a closed subforest. The yellow part does not correspond to a closed subforest since the subtrees are not consecutive siblings.*

4.7 Visualization of RNA Secondary Structure Alignments

4.7.1 ASCII Representation

The ASCII representation of an RNA secondary structure alignment extends the sequence alignment representation that arranges the aligned sequences on top of each other. Essentially, it is an alignment of Vienna strings (see Section 2.3) and there is a gap in the structure line iff there is a gap in the sequence line. See the following example where a "*" character highlights sequence or structure conservation:

Input

```
> alanine
ggggcuauagcucagcugggagagcgcuugcauggcaugcaagaggucagcgguucgaucccgcuuagcuccacca
(((((((..((((........)))).(((((.......))))).....(((((.......))))))))))))....
> leucine
gccgaaguggcgaaaucgguagacgcaguugauucaaaaucaaccguagaaauacgugccgguucgaguccggccu
(((((((..(((...........))).(((((.......))))).(((....)))..(((((.......)))))))
ucggcacca
)))))....
```

Output

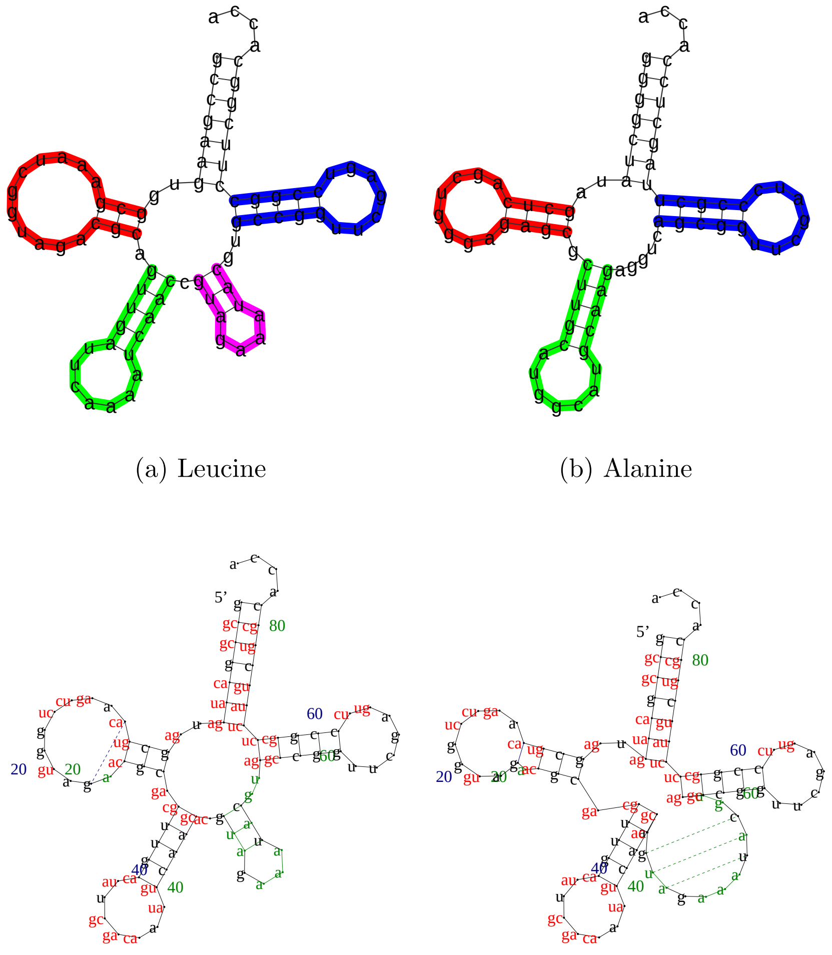
```
alanine      ggggcuauagcucagcugggag-agcgcuugcauggcaugcaagag--g---u-c
leucine      gccgaaguggcgaaaucgguagacgcaguugauucaaaaucaaccguagaaauac
             *  *   * **  *   ** **  **  ***  *   *  ***  *  *   * *
alanine      --agcgguucgaucccgcuuagcuccacca
leucine      gugccgguucgaguccggccuucggcacca
                 ********  ***     *  *****

alanine      (((((((..((((........)-))).(((((.......)))))..--.---.-.
leucine      (((((((..(((...........))).(((((.......))))).(((....)))
             ************ ********  **********************   *
alanine      --(((((.......))))))))))))....
leucine      ..(((((.......))))))))))))....
               ****************************
```

4.7.2 2d-Plot

RNA secondary structures are represented graphically as circle plots, dot plots, mountain plots or 2d-plots (refer to Section 2.3). I present a 2d-plot variant for RNA secondary structure alignments that emphasizes both sequence and structure similarity. I follow the strategy of using well established layout algorithms for 2d-plots of RNA secondary structure [14, 109, 143, 179, 234][5]. Therefore, I derive a secondary structure from a structure alignment which is drawn and annotated further. Since bases paired in a structure S_1 can be aligned to bases unpaired in a structure S_2, the presentation of a common secondary structure leaves some choice. For an alignment A of structures S_1 and S_2, I draw an RNA secondary structure "S_2-at-S_1" that highlights the differences as deviations of S_2 from S_1, or vice versa "S_1-at-S_2". Both are alternative visualizations of the same alignment A. The drawings can be annotated using all the information of the alignment, e.g. show alternative base pairings as dashed lines connecting bases.

Figure 4.10 explains the visualization by an example. The visualization of local similarity follows the same strategy. If suboptimal local alignments are calculated, the local similar regions are highlighted in the original structures. Figure 4.13 shows local similar regions of the structures in Figure 4.11 and Figure 4.12. I am sure to make comparing RNA secondary structures quite comfortable by using this visualization.

[5] I use an implementation of Bruccoleri et al.'s *NAVIEW* algorithm [14].



(a) Leucine

(b) Alanine

(c) Alanine-at-Leucine

(d) Leucine-at-Alanine

Figure 4.10: *Secondary structures of E.coli tRNA for leucine (Anticodon CAA) (a) and alanine (Anticodon GGC) (b), taken from the Genomic tRNA Database [116]. 2d-plot of the structure alignment of tRNAs for Alanine-at-Leucine (c) and Leucine-at-Alanine (d). The acceptor stem (red), anticodon stem (green) and TψC stem (blue) have the same length in both structures, but some differences with regard to the sequence. There are also sequence variations in the single-stranded regions, especially at the anticodon position. The visualization emphasizes this automatically by using red letters. For the double-stranded regions an accentuation of compensatory base exchanges is achieved by this presentation. Bases printed in black show structure elements that occur in both structures with the same sequence. The CCA at the 3' end is a typical invariant feature of tRNAs and so its printed in black. Structural elements, which can only be found in the first structure are printed in blue. Thus, the fourth base-pair in the D-stem (magenta) of tRNA for alanine is shown with the dashed blue line in the alignment. In contrast, structural elements shown in green occur only in the lysine structure. The extra stem of the leucine tRNA is highlighted that way.*

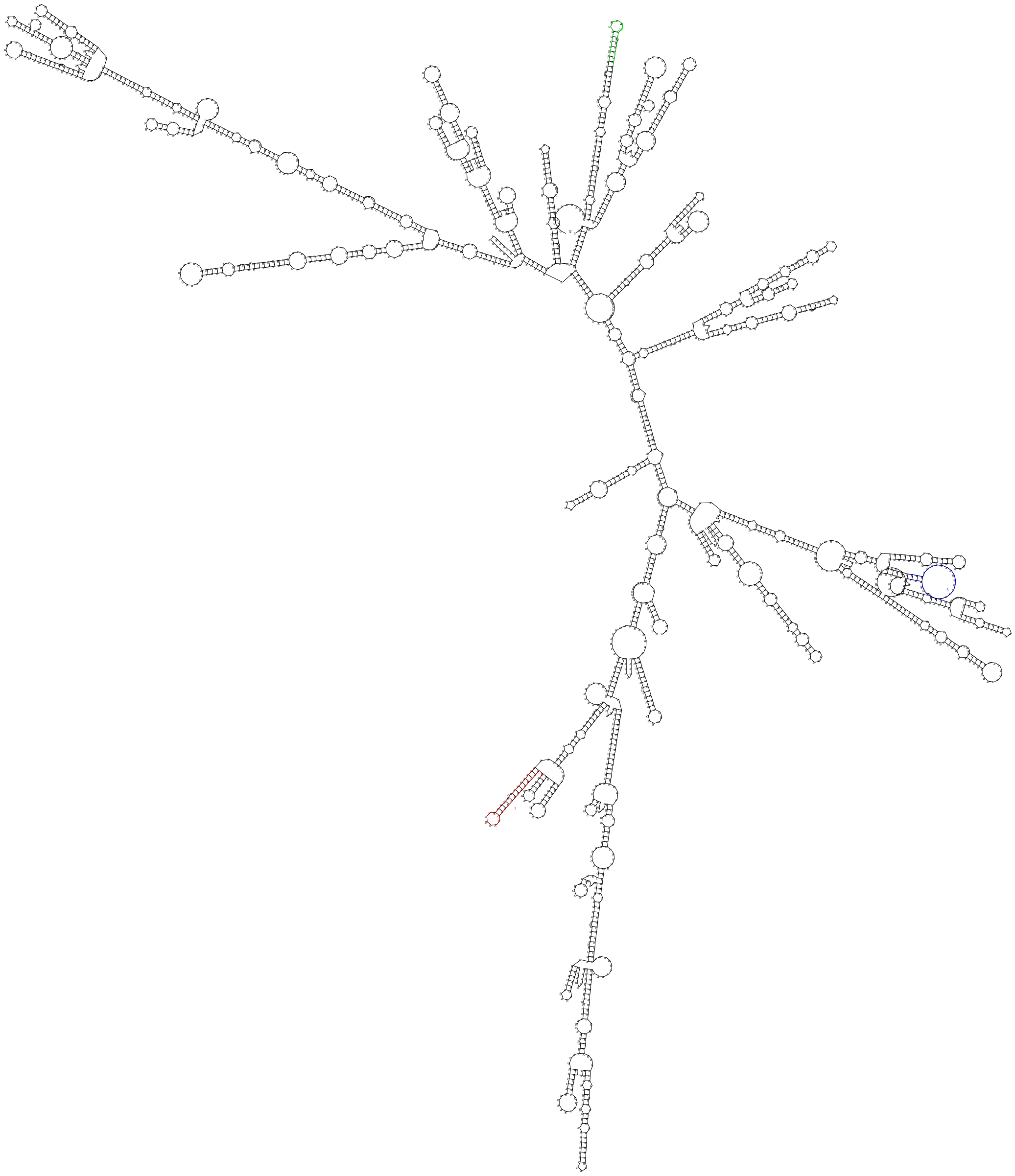


Figure 4.11: *Predicted structure of the human transferrin receptor 3' UTR. The sequence was taken from the UTR database, accession number 3HSA008842 [154]. The colored regions highlight the local similar parts to the structure in Figure 4.12.*

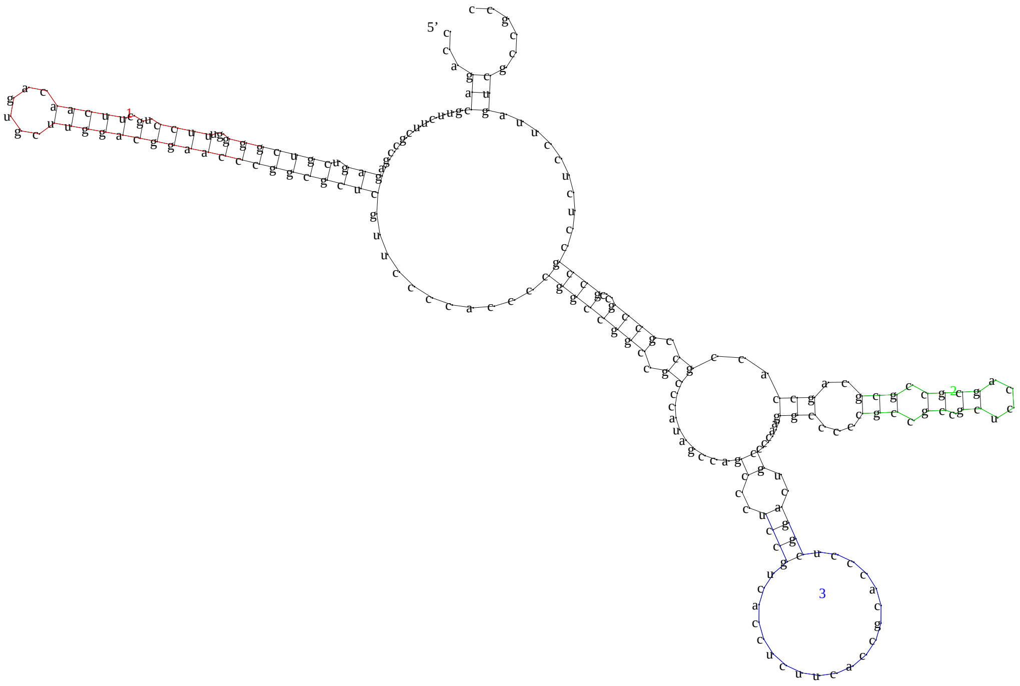
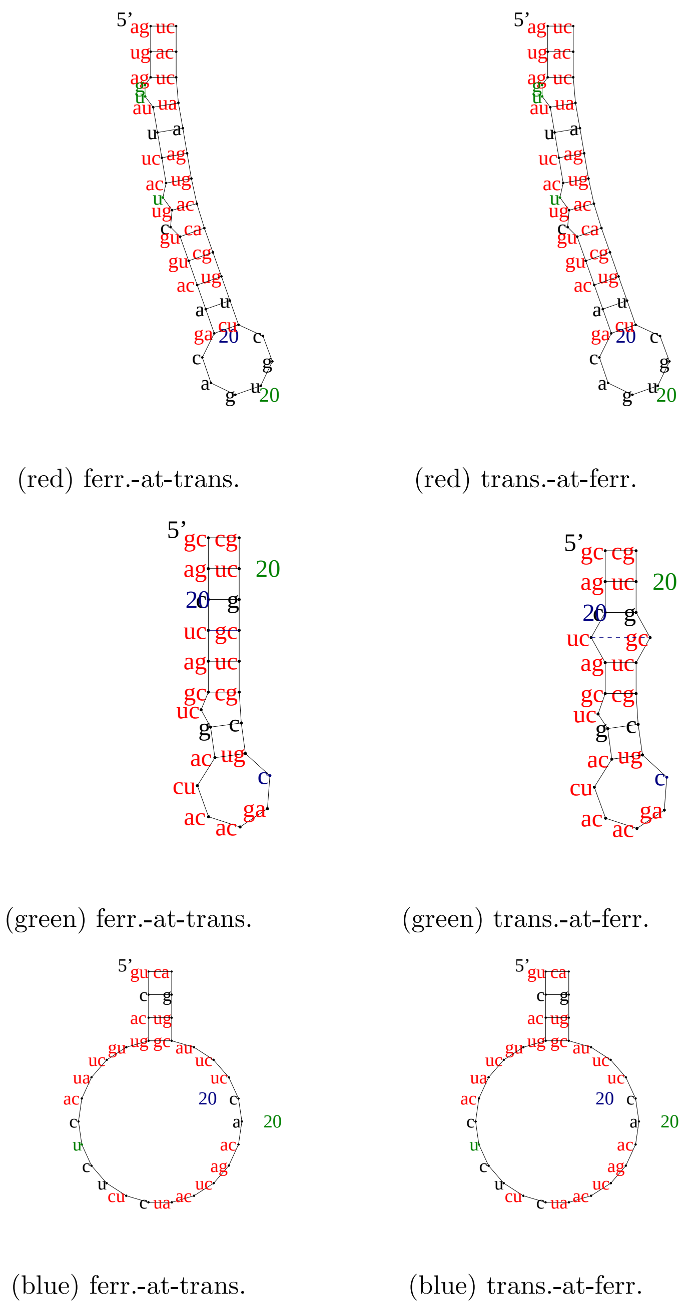


Figure 4.12: *Predicted structure of the human ferritin 5' UTR. The sequence was taken from the UTR database, accession number 5HSA015337 [154]. The colored regions highlight the local similar parts to the structure in Figure 4.11.*



(red) ferr.-at-trans.

(red) trans.-at-ferr.

(green) ferr.-at-trans.

(green) trans.-at-ferr.

(blue) ferr.-at-trans.

(blue) trans.-at-ferr.

Figure 4.13: *Local alignments of the human transferrin receptor 3' UTR (Figure 4.11) and the human ferritin 5' UTR (Figure 4.12). (red) shows the best scoring local alignment which is found at the positions 932 in transferrin and position 26 in ferritin. This motif is the well studied* Iron Responsive Element*(see 7.1). (green), and (blue) show suboptimal local alignments that were found in the structures. These were found at the positions* 2392 *and* 147*, and* 1765 *and* 104*, respectively. As I focus on the visualization technique here, the putative biological function of these regions is not further discussed.*

4.8 Performance of Forest Alignment Algorithms

In Section 4.8.1, I analyze the parameters of an RNA secondary structure that affect the complexity of the algorithms presented in this chapter. In Section 4.8.2, I provide measurements concerning the practical runtime and space requirement of my algorithms.

4.8.1 Efficiency Considerations for RNA Secondary Structure Alignments

The efficiency analysis of Algorithm 4.1 in terms of RNA secondary structures requires to observe which parameters of an RNA secondary structure affect the size and the degree of a forest in the extended forest representation.

The total number of forest nodes consists of the sum of P-nodes and B-nodes. The number of B-nodes equals the sequence length. Each P-node "consumes" two B-nodes and, thus, there are at most half as many P-nodes as B-nodes. Hence, in extended forest representation, the number of tree nodes grows linear with the sequence length. The length of unpaired regions and the branching degree of multiloops determine the degree of a forest. For RNA secondary structures that exist in nature the maximum length of an unpaired region can be considered to be bounded by some constant. For reasons of thermodynamic stability, loops cannot be arbitrary large. Hopefully, above a certain sequence length, the number of branches in a multiloop can also be considered to be constant or at least to grow slowly. I now turn to measure these parameters.

4.8.2 Measurements

For the following experiments, I generated a sample set of RNA secondary structures which consists of predicted RNA secondary structures for sequences ranging from length 5 to 1000. For each sequence length, 10 se-

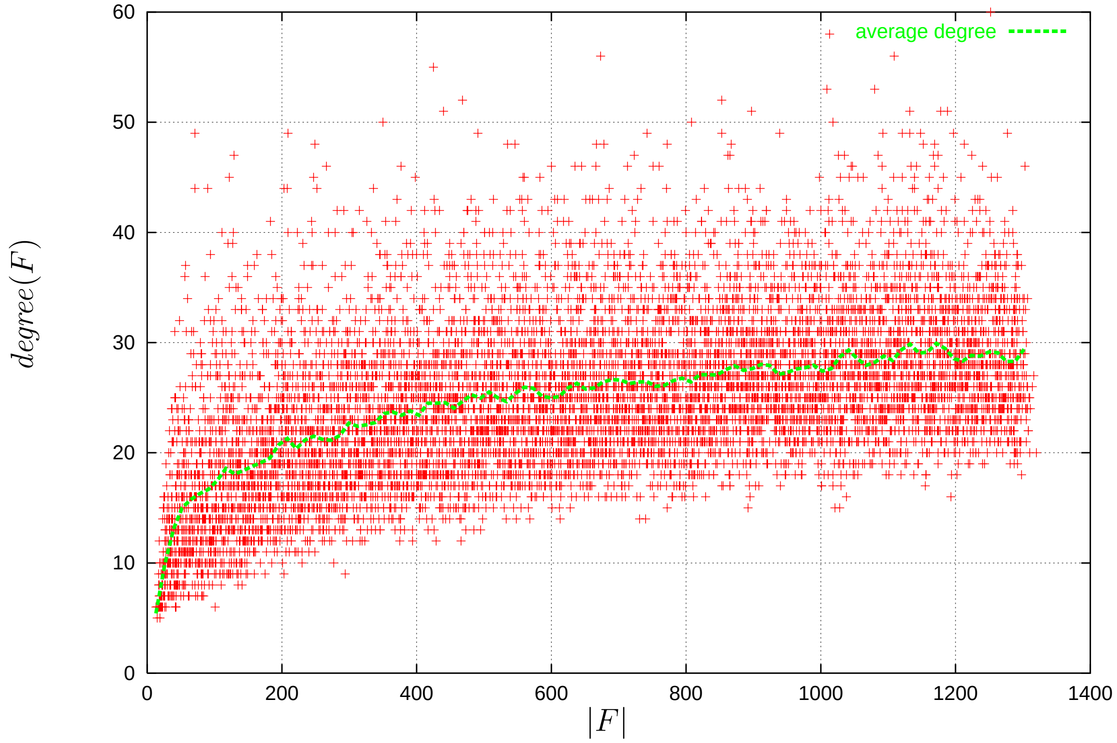


Figure 4.14: *Degree measurements on forests in the extended forest representation for RNA secondary structures generated from folded random sequences: The degree of a forest is plotted against the number of nodes.*

quences are generated assuming an equal distribution of the bases A, C, G, U. The prediction was done using *RNAfold* [84]. The complete dataset includes 9960 structures. I argue that the results show the worst case when analyzing real RNA structures (or real sequences that are folded). A structure that occurs in nature should obtain, if at all, slightly better energy values and contain smaller loops. The following measurements were done for the extended forest representation of RNA secondary structures.

In Figure 4.14, I plot the degree of a forest against the number of nodes. It turns out that the degree of a forest converges to an average value of approximately 30. The maximum degree measured is 60. This shows that the practical runtime can be expected to be quadratic in the number of nucleotides in a structure.

In Figure 4.15, I plot the depth of a forest against the number of nodes.

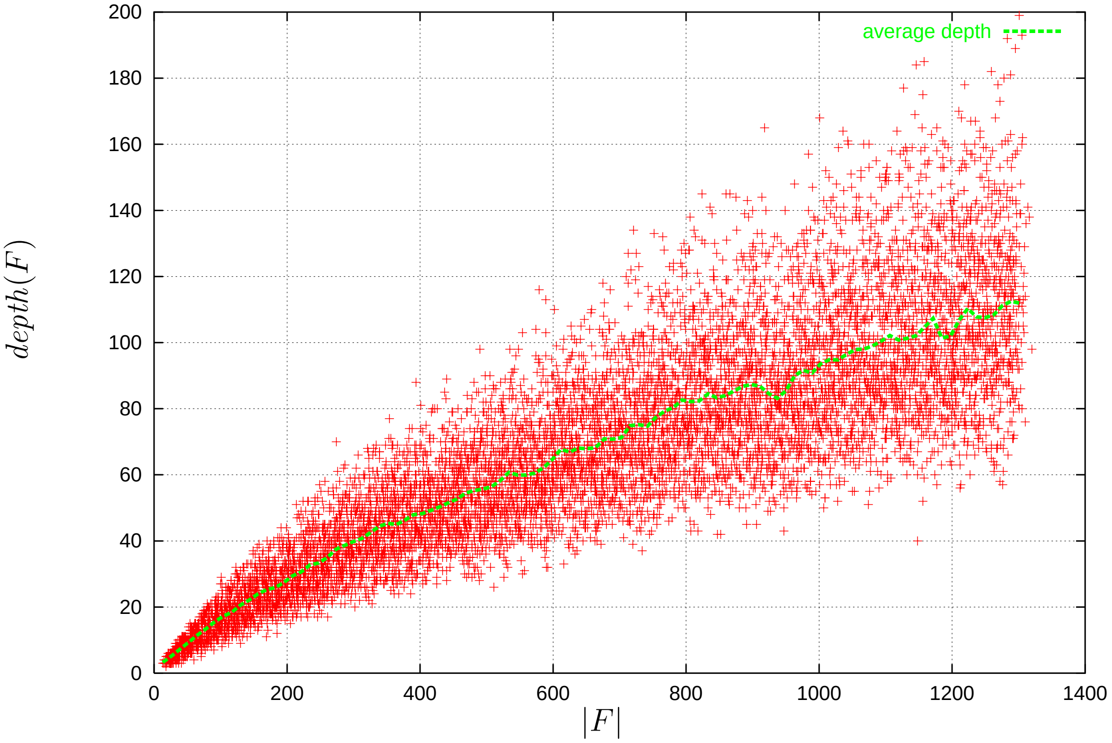


Figure 4.15: *Depth measurements on forests in the extended forest representation for RNA secondary structures generated from folded random sequences: The depth of a forest is plotted against the number of nodes.*

The depth of a forest is not relevant for the forest alignment algorithms but is crucial for the tree edit algorithms (refer to Section 2.5.3). In contrast to the degree, the depth does not converge, but seems to grow sub-linear. Hence, considering the proposed algorithms for computing the tree edit distance, the practical runtime of RNA secondary structure comparison in the tree edit model is more than quadratic. This result is consistent with the theoretically derived average runtime for the Zhang-Shasha tree edit algorithm which is $O(|T_1|^{\frac{3}{2}} \cdot |T_2|^{\frac{3}{2}})$ [39]. I assume that the average of the measured degrees approximates some root function.

For space complexity, the combined measure of the number of nodes and the degree is the number of closed subforests of a forest. The square of this number determines the size of the tabulation matrix, and the number of relevant closed subforests for the local closed subforest similarity. Figure 4.16

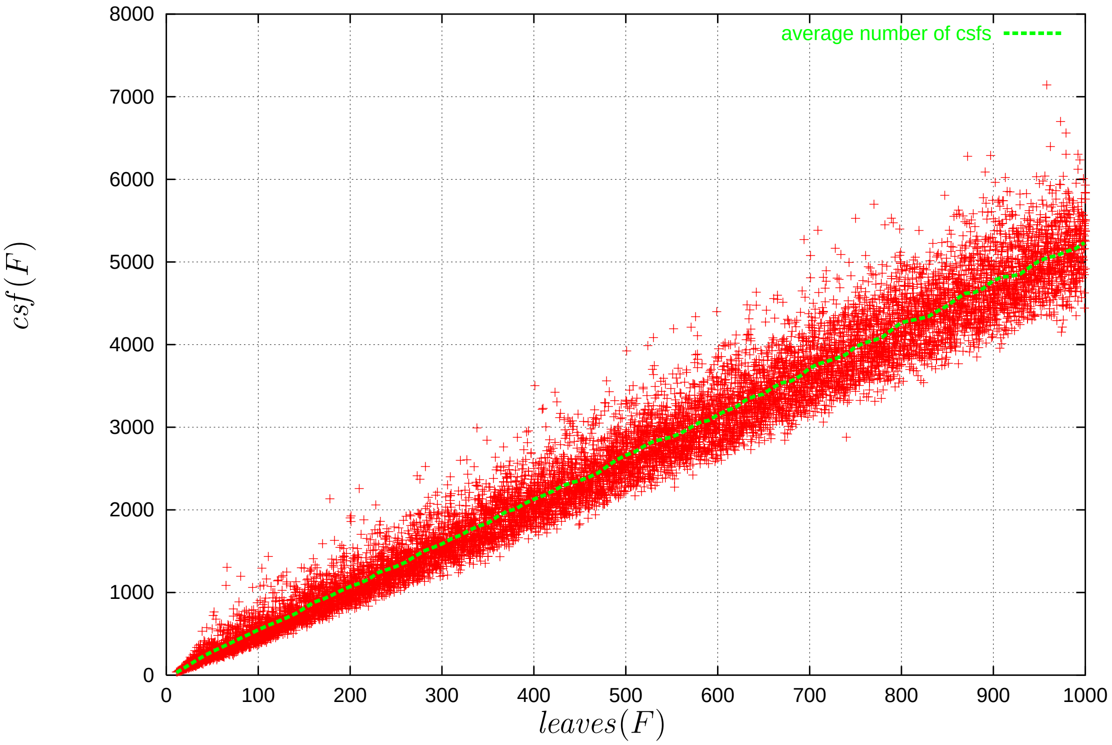


Figure 4.16: *The number of closed subforests for forests in the extended forest representation for RNA secondary structures generated from folded random sequences: The number of closed subforests is plotted against the number of leaves (or equivalently the sequence length).*

shows how the number of closed subforests scales depending on the sequence length. It turns out that the number of closed subforests grows linearly and, thus, the practical space complexity of Algorithm 4.1 is quadratic. Figure 4.17 shows the space consumption in Megabyte for the 4-dimensional (without the second stage mapping β, see Section 3.3.2) and the 2-dimensional tabulation, respectively. Figure 4.18 shows the percentage of space that the 2-dimensional tabulation consumes in comparison to the 4-dimensional tabulation.

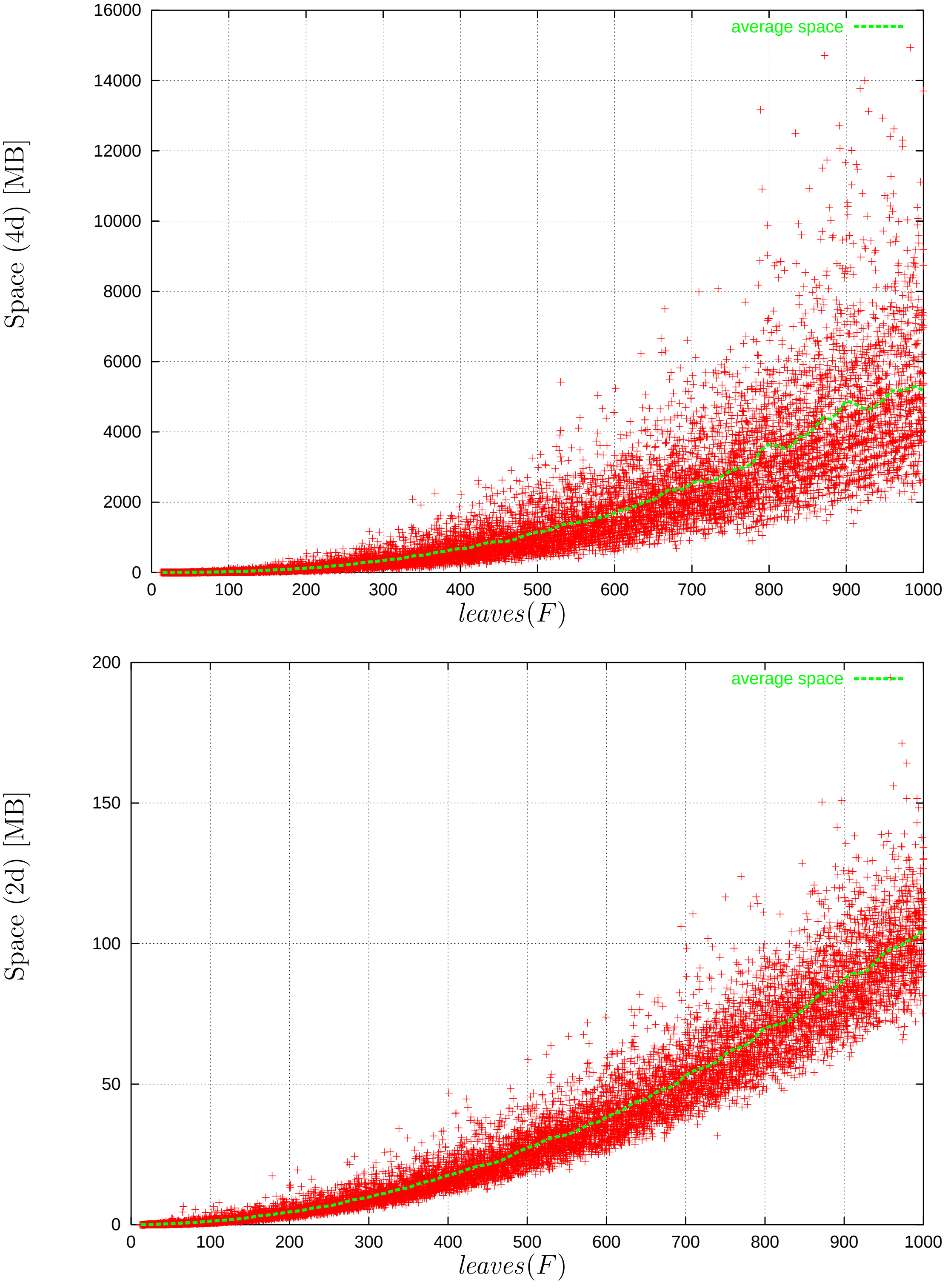


Figure 4.17: *Space requirement of the 2- and 4-dimensional tabulation for two forests of the same size (meaning that they have the same number of closed sub forests).*

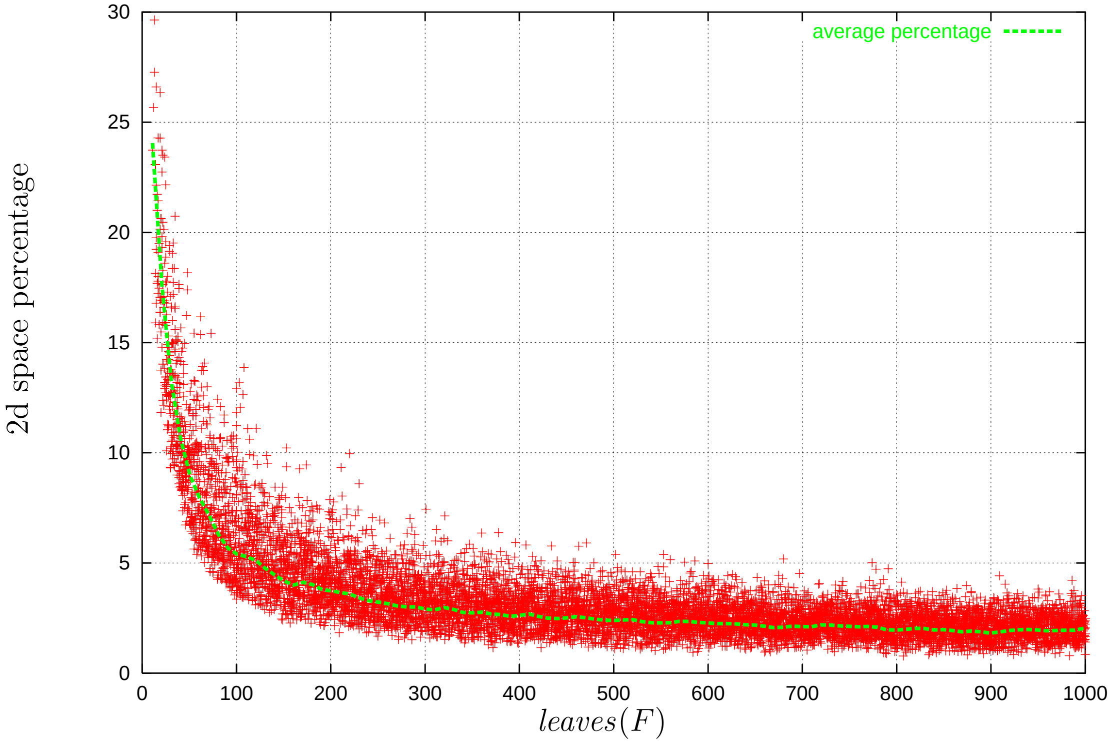


Figure 4.18: *The space improvement of the 2-dimensional tabulation is measured in terms of the percentage of space that the 2-dimensional tabulation consumes in comparison to the 4-dimensional tabulation.*

The time measurements consider the practical runtime of the global and local welformed forest alignment model. For both variants, I distinguish between algorithms that implement the reduced search space recurrences (see Section 3.3.2) and those that do not. The calculations were done on a *SunFire V60x* with 2GB RAM, Intel Xeon CPUs (2 x 2.8 GHz), and Solaris 10 operating system. All algorithms are implemented in the tool *RNAforester* (see Section 6) which was used for these measurements. Figure 4.19 shows the measured time for the calculation of global and local similarity depending on the sequence length. The reduced search space variant of global and local similarity gives a constant speedup of approximately 2.5.

All these measurements document that my algorithms have time and memory demands that are suitable for the analysis of real data, even in large scale applications.

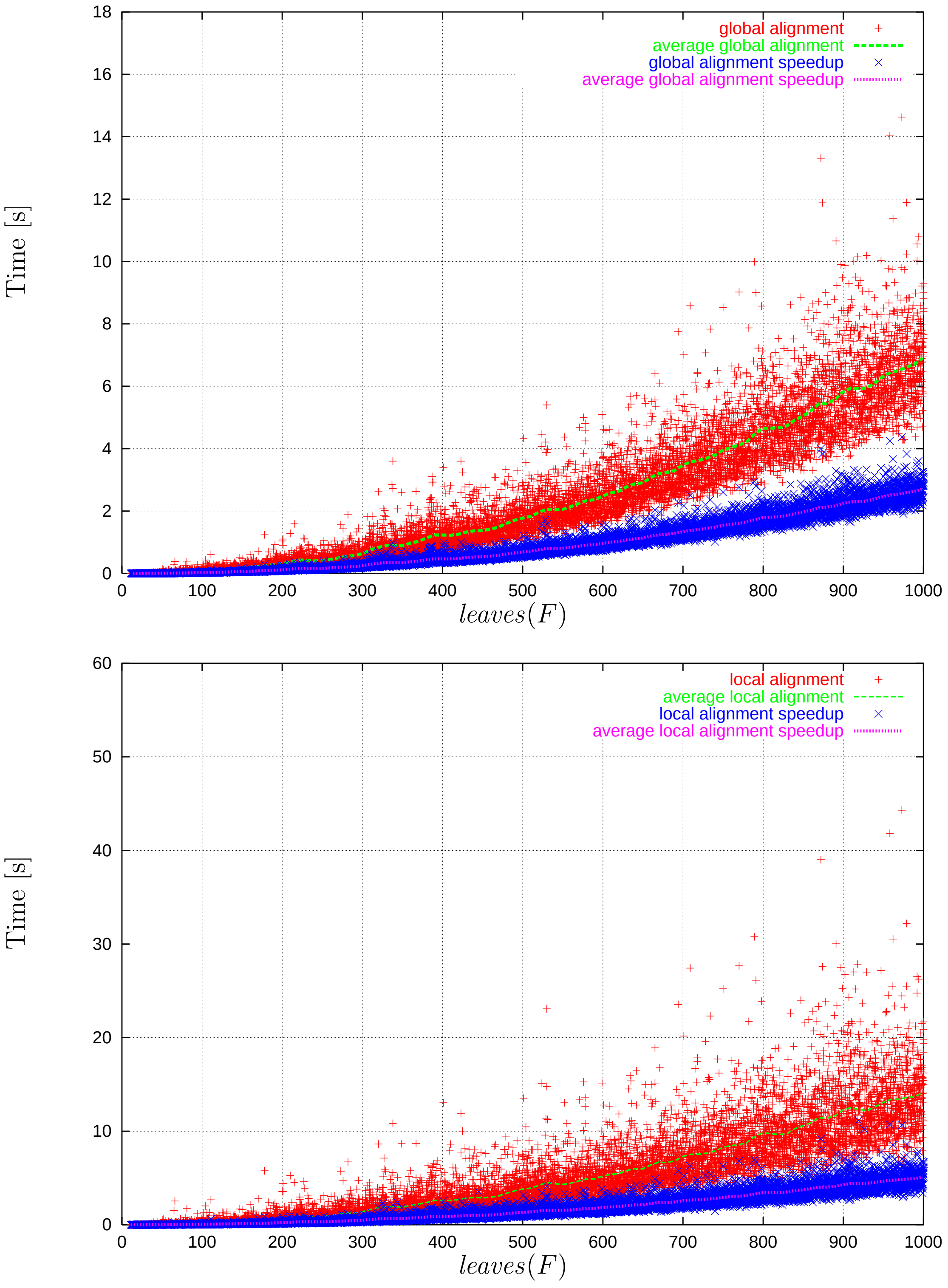
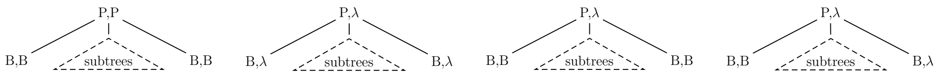


Figure 4.19: *Time in seconds for the calculation of global and local forest alignments with and without the reduced search space recurrences due to Section 3.3.2.*



(a) base-pair replacement

(b) base-pair deletion

(c) base-pair breaking

(d) base-pair altering (right)

Figure 4.20: *(a)-(d) show how the alignment structure is related to edit operations.*

4.9 The Welformed Forest Alignment Model Revisited

Continuing the uniform description of edit based models for RNA structures in Section 2.5.5, I analyze the welformed forest alignment model under the scope of Jiang et al.'s general edit model for RNA structures.

The classical sequence edit operations affect the leaf nodes in the extended forest representation (B-nodes). The structural edit operations are assembled from edit operations on P-nodes and B-nodes: The base-pair breaking corresponds to the deletion of a P-node. The base-pair deletion is modeled by the independent deletions of a P-node and the two paired B-nodes. The corresponding holds for the base-pair replacement and the base-pair altering edit operation. Figure 4.20 shows how the forest alignment model is related to the described edit operations.

edit operation	score
base replacement	$\sigma(B, B)$
base indel	$\sigma(B, \lambda)$ and $\sigma(\lambda, B)$
base-pair replacement	$\sigma(a{\circ}b, c{\circ}d)$ or $\sigma(P, P) + 2 \cdot \sigma(B, B)$
base-pair breaking	$\sigma(P, \lambda)$ and $\sigma(\lambda, P)$

In terms of edit operations, the welformed alignment model is closest to Bafna et al.'s model. Their model also builds structural edit operations from base-pair breaking and base replace and indel operations. However, there is a substantial difference between the welformed alignment model and the edit models presented in Section 2.5.5: The alignment model is not an operational model. A model consisting of the above operations does *not* correspond to the welformed alignment model. The following example elucidates this: Consider a scoring contribution of zero for the relabeling, insertion, and deletion of a P-node. In this case, an operational model corresponds to the classical sequence alignment model, i.e. an optimal score can always be obtained by first deleting all P-nodes. This does not hold for the forest alignment model where relabeling, insertions, and deletions must be consistent with the forest structure. Refer to Section 2.5.3 for the properties of tree/forest alignment models.

Chapter 5

Multiple Alignment of RNA Secondary Structures

In the world of biomolecular sequences, the multiple sequence alignment is an ubiquitous, indispensable means to reveal the traces left by the evolution of a group of related nucleic acid sequences. The calculation of multiple sequence alignments has become a discipline on its own in Bioinformatics and numerous publications exists in this still highly active field. Surveys are provided in [41, 63, 148]. Multiple sequence alignment tools, among the most popular ones are *ClustalW* [200], *DiAlign* [140, 141], *Prrp* [62], *MSA* [115], *DCA* [186] and *T-Coffee* [149], find brisk application in phylogenetic analyses, the identification of conserved motifs, and domain and structure prediction. These applications are interesting for amino acid sequences as well as nucleic acid sequences. Their success depends largely on the quality of the multiple alignments. The quality in turn depends on the level of sequence conservation.

Structural RNA can exhibit low sequence conservation, though the secondary structure is highly conserved, i.e. the same structure can be formed by different sequences with that same base-pair pattern. Thus, bases that form base-pairs are expected to be less sequence conserved than unpaired bases after some (for a human life-span incredibly large) time of evolution.

This has consequences for the above applications for structural RNA:

- After some point in time in the evolution of structural RNA, the sequence diversity is no longer a measure for the evolutionary distance between sequences; sequences are just dissimilar. The selective pressure is on the structural level and, though the sequence changes permanently during evolution, the structure remains similar. Accordingly, a distance on structures should be a better measure to construct phylogenies for distantly related structural RNA molecules.

- The identification of conserved motifs is limited to the identification of sequence motifs. If a motif is structurally conserved, a biological correct alignment of the motif depends on the conservation of the sequences that surround it. Homologous sequence regions are matched in the alignment and force regions of sequence diversity between them to align to each other. If the surrounding parts are sufficiently sequence conserved, there is a chance to align the "non-fitting" structural parts correctly. If not, the method will fail. Moreover, the detection of structural conserved regions requires additional effort since the sequence alignment score for these regions does not identify them. A multiple alignment that considers both sequence and structure would eliminate these problems.

- The structure prediction of RNA molecules by a multiple sequence alignment has the same intrinsic problem as the identification of structural motifs. If sequences are not sufficiently conserved, the structural regions cannot be aligned correctly.

It is obvious that additional structural information can improve the quality of a multiple alignment of structural RNA. In the following, the multiple forest alignment model is considered. I propose a notion of consensus structures for a family of RNA molecules, the *RNA secondary structure profiles*, and provide intuitive visualizations for them. To demonstrate the usefulness of a multiple RNA secondary structure alignment, I propose a consensus structure prediction strategy for families of RNA molecules that have low sequence homology. I have published central ideas of this Chapter in [78].

Related work has been done by Torsello et al. [204, 206]. They considered the clustering and classification of shape abstracted images which are represented as skeletal trees. Similar to my approach, the clustering is done by merging trees to get a kind of representative for multiple trees. I will further comment on their approach in Section 5.4.

5.1 Multiple Alignment Strategies

The exact calculation of multiple sequence alignments is extremely demanding for computer resources. The complexity of this problem depends on the scoring function. Among different scoring schemes, the sum-of-pairs score is the one that received most attention [15]. Wang and Jiang showed that the problem of computing a multiple sequence alignment with optimal sum-of-pairs score is NP-complete [220]. This remains true if the scoring scheme is a metric one [12]. Therefore, one cannot hope to compute the edit or alignment distance (or similarity) exactly within polynomial time. Driven by the importance of multiple sequence alignments in the field of molecular biology, several heuristics and approximation schemes have been developed that produce "good" alignments. Essentially, there are two general ideas how to produce near optimal multiple sequence alignments, the progressive strategy and the simultaneous strategy:

- **Progressive strategy:** A progressive alignment is sometimes also referred to as an *iterative alignment*. However, there is no strict convention about this terminology and the term iterative alignment is also used for the following strategy: A multiple alignment is constructed, by whatever heuristic, and refined through a series of iterations until no further improvements can be made. A genetic algorithm approach as implemented in the tool *SAGA* is an example of an iterative strategy [150].

In a *progressive* strategy, the calculation of a multiple alignment is reduced to an iterated application of pairwise alignments. This requires a concept to align alignments or join a sequence to an existing alignment. The first description of a progressive algorithm is due to Hogeweg & Hesper [88]. *ClustalW* is the most prominent implementation of a progressive algorithm [46]: From the pairwise comparison of all combinations of sequence pairs, a guide tree is constructed by hierarchical clustering. The multiple alignment is then built from pairwise alignments along this guide tree. Additionally, *ClustalW* includes features such as affine gap penalties, automatic substitution matrix choice or the automatic gap penalty adjustment to improve the quality of the multiple alignment.

- **Simultaneous strategy:** Another way to speed up the calculation of multiple alignments is to reduce the problem size. Larger sequences are divided into smaller sequences and those are aligned. Afterwards, the whole alignment is built by merging the smaller alignments. This is the general idea of *divide and conquer algorithms*. The difficulty is to cut the sequences at the correct point. Among others, the multiple alignment tools *Prrp* [62], *DCA* [186] and *DiAlign* [140, 141] follow this strategy. The latter is also progressive, since the global alignment is constructed by progressively arranging highly similar regions of the sequences.

Alignment of Multiple RNA Secondary Structures While multiple sequence alignment strategies are getting more and more sophisticated and specialized, multiple RNA secondary structure alignment strategies are just at the beginning. Driven by the now generally acknowledged importance of structural RNA, new approaches were proposed recently: Interactive tools like *ESSA* [24], *ConStruct* [120] and *Stemtrace* [231] analyze conserved patterns and consensus structures by combining thermodynamic and comparative methods. These tools allow and often require manual intervention.

Thus, they are not suitable for analyzing large data sets automatically. However, they are extremely helpful in refining the results of computational approaches.

Siebert & Backofen provided a multiple alignment strategy for structural RNA based on the multiple sequence alignment tool *T-Coffee* [149, 183]. *T-Coffee* optimizes a multiple alignment according to a library of pairwise alignments. In their multiple alignment tool *MARNA*, Siebert & Backofen calculate pairwise alignments using Zhang et al.'s method [242]. Since *T-Coffee* computes a sequence alignment, the problems that were reported for sequence alignment strategies in Section 2.5.2 cannot be avoided.

Hofacker et al. provided a strategy that is based on aligning base-pair probability matrices, predicted by McCaskill's partition function algorithm [80]. Their strategy is in flavor of Sankoff's algorithm [172] and is implemented in the tool *pmmulti* of the Vienna RNA package.

Wang & Zhang generalized Zhang et al.'s structural alignment model for more than two structures in a progressive fashion [222]. Their model lacks the base-pair breaking operation (refer to Section 2.5.5), which limits the quality of structural alignment especially on the sequence level.

In the following, I will provide a multiple RNA secondary structure alignment algorithm based on the forest alignment model.

5.2 Multiple Alignment of Forests

In Section 2.5.3, I reviewed the tree editing distances and gave alternative formulations of the problems in terms of edit sequences, mappings and graph isomorphisms. Here, I consider the extension for the multiple, not necessarily pairwise, case. A natural extension of the tree edit and tree alignment distance is motivated by the graph isomorphism definitions of these problems. The tree edit model considers isomorphic subforests, while the tree alignment model considers isomorphic supertrees. This concept can be extended directly to an arbitrary number of trees and forests. I concentrate on the alignment model for the following reasons:

- An alignment of forests is a forest and, hence, can again be aligned in the forest alignment model. This makes virtually every progressive strategy that was reported for multiple sequence alignment applicable to multiple forest alignments. The idea of the algorithms persists even if the type of the alignment is not a sequence. Remember that a mapping between forests is not necessarily consistent with the forest structure and the generalization of the edit based approach would require the definition of a ”multi-mapping”.

- Based on the observations in Section 4.8.2, I expect to achieve a better practical runtime using the alignment model.

- A multiple forest alignment is a compact data structure that is suitable to represent a family of RNA structures. The concept of sequence profiles can be naturally extended to forests, which results in a profile representation of RNA secondary structures.

I now turn to formalize the multiple forest alignment model. Consider the definition of function π in Equation (2.10), an alignment of n forests is defined as follows:

$$A \in \mathcal{F}(\Sigma_\lambda^n) \textit{ is an alignment of forests } F_1, F_2, \ldots, F_n \in \mathcal{F}(\Sigma) \text{ iff} \\ F_i = \pi(A|_i) \text{ for } i \in [1, n]. \quad (5.1)$$

Note that the labels of a multiple forest alignment are n-tuples. Figure 5.1 shows an example of an alignment of four RNA secondary structures in the extended forest representation. As an optimization criterion, a scoring function for multiple forest alignments is required. The sum-of-pairs score introduced by Carillo & Lipman [15] defines the score of each column of a multiple sequence alignment as the sum of scores of all combinations of pairwise scores for the column. Let n be the number of columns which corresponds to the number of aligned forests in a forest alignment A. The *sum-of-pairs (SP) score* is defined formally as:

$$\sigma_{\mathrm{SP}}(A) = \sum_{v\ node\ in\ A} \sum_{1 \leq p \leq q \leq n} \sigma(label(v)_p, label(v)_q). \tag{5.2}$$

As I represent RNA secondary structures in the extended forest representation, I concentrate on the welformed alignment similarity σ_{WFA} (see Section 4.3). The definition of welformed forest alignments that was given in the context of pairwise alignments (see Definition (4.1)) applies to the multiple case as well. I define the alignment similarity σ_{WFA} between forests $F_1, F_2, \ldots, F_n$ as the maximum SP-score that a welformed alignment of $F_1, F_2, \ldots, F_n$ can achieve[1].

$$\sigma_{\mathrm{WFA}}(F_1, F_2, \ldots, F_n) = \max\{\sigma(A) \mid \text{A is a welformed alignment of } F_1, F_2, \ldots, F_n\}. \tag{5.3}$$

5.3 A Forest Profile for RNA Secondary Structures

Multiple alignments of protein sequences are useful to group proteins of similar functions into protein families. The identification of proteins that also belong to a certain family gives naturally rise to the question of finding a kind of representative sequence for a protein family. Such representations, that are well known for multiple sequence alignments, are *profile* and *consensus sequence*. Here I restrict my attention to the profile representation [66]. A *profile* for a multiple sequence alignment consists of the frequencies of characters in each row and is also known as a *weight matrix*.

In analogy to view sequence alignments as sequences of edit operations,

[1] The generalization of the classic forest alignment model follows analogously.

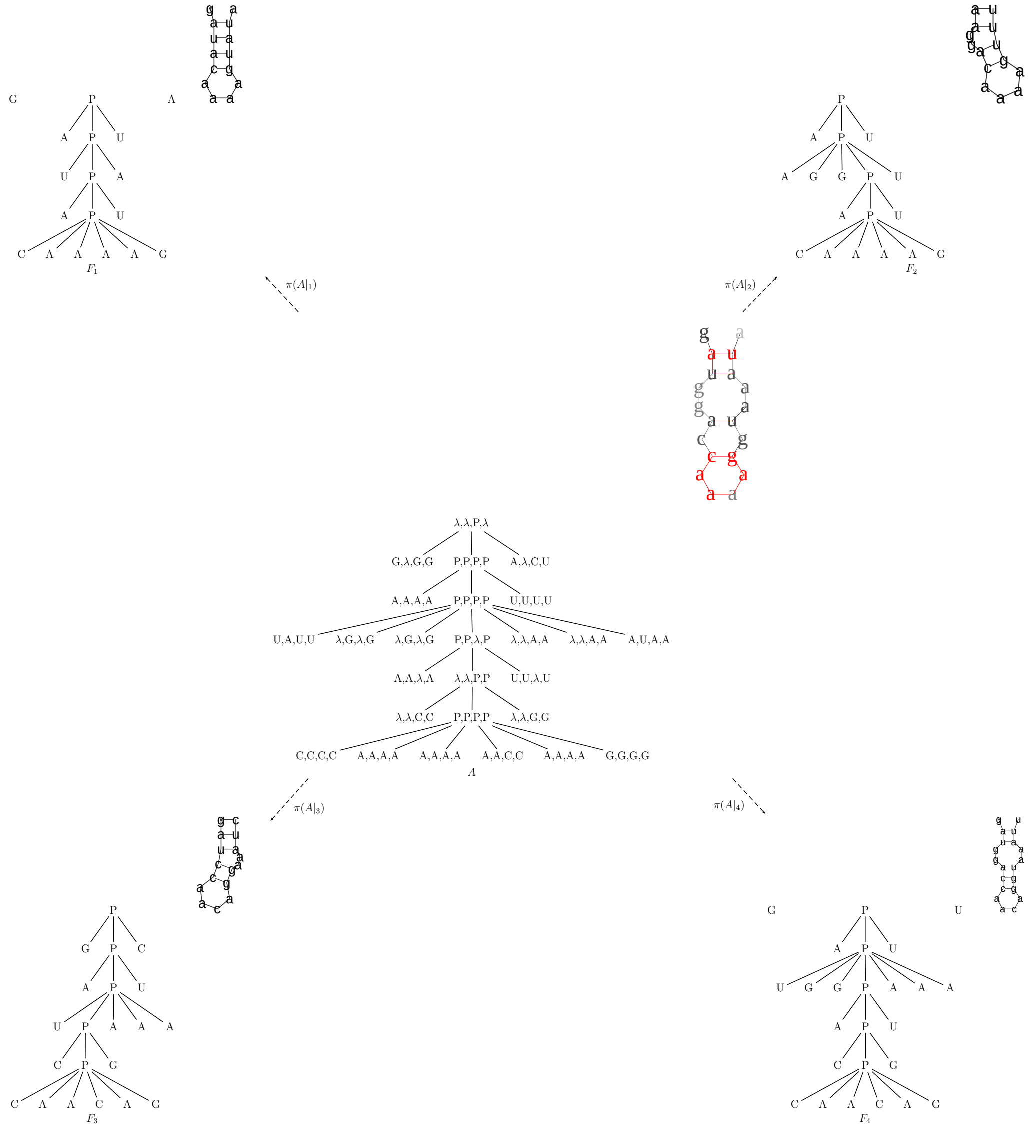


Figure 5.1: *A is an alignment of F_1, F_2, F_3, F_4. The 2d-plot of the secondary structure is shown at the top right corner of the forests. The 2d-plot for the alignment A will be explained in Section 5.3.1. Intuitively, it shows an overlay of the single structures.*

and tree alignments as trees labeled with edit operations (see Section 2.5.3), I consider a profile for a sequence alignment as a sequence of relative frequency vectors. Consequently, a profile for a forest alignment is a forest labeled with relative frequency vectors. Let $k = |\Sigma \cup \{\lambda\}|$, I give the following definition of a *profile for a forest alignment*:

Given a multiple forest alignment $A \in \mathcal{F}(\Sigma_\lambda^n)$, its profile alignment $P_A \in \mathcal{F}(\mathbb{R}^k)$ is obtained by converting each label in A to its relative frequency vector. (5.4)

An example of a multiple forest alignment and its corresponding profile is shown in Figure 5.2. Since a profile for a forest alignment is also a forest, it is straightforward to define a *profile-forest to profile-forest alignment.*

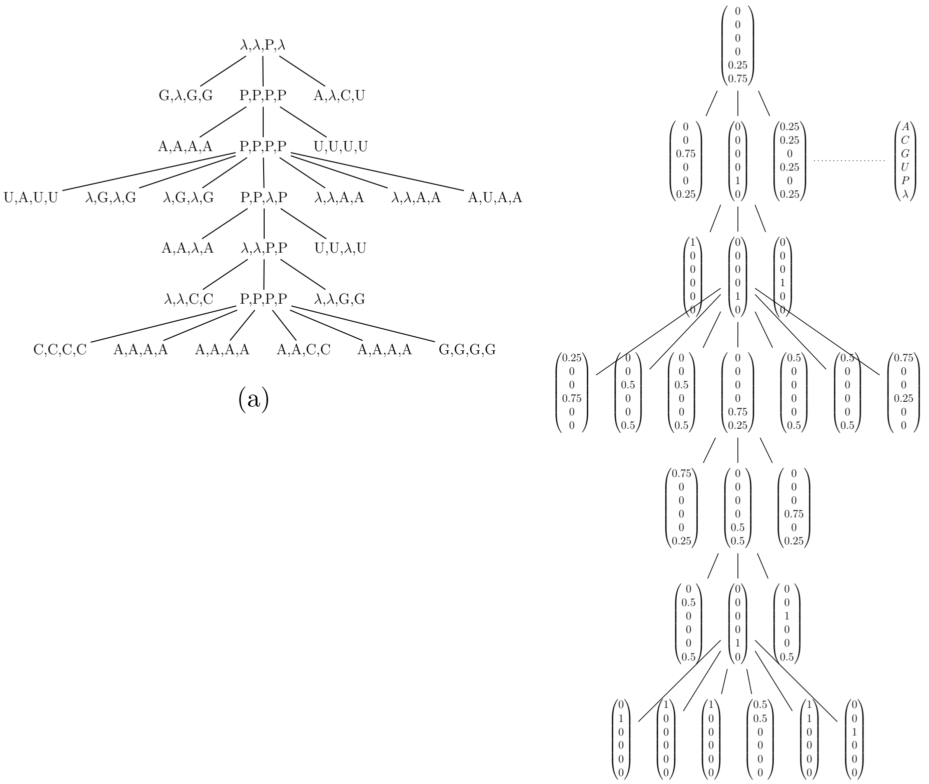
Adhering to the sequence alignment tradition, I use the analog scheme to the sum-of-pairs score for frequency vectors. The *profile sum-of-pairs score* σ_{SP} of two relative frequency vectors $p, q \in \mathbb{R}^k$ is defined as follows:

$$\sigma_{\text{SP}}(p, q) = \sum_{(a,b) \in \Sigma_\lambda^2} p_a \cdot q_b \cdot \sigma(a, b). \tag{5.5}$$

Unlike for distances where the score of two equal forests is zero, the similarity value can be an arbitrary positive value. The similarity score of two equal forests of size n can be the same as for two different forests of size m where $m > n$. Therefore, I introduce relative scores that are upper bounded by 1. The *relative similarity score* $\sigma_{\text{SP\_REL}}$ of forests F_1 and F_2 is defined as:

$$\sigma_{\text{SP\_REL}}(F_1, F_2) = \frac{2 \cdot \sigma_{\text{SP}}(F_1, F_2)}{\sigma_{\text{SP}}(F_1, F_1) + \sigma_{\text{SP}}(F_2, F_2)} \tag{5.6}$$

The self-similarity score of a forest results from a perfect matching alignment for reasonable scoring schemes. This score can be computed, without self-aligning the forests, in $O(|F|)$. It is simply the sum of the self-relabeling



(a)

(b)

Figure 5.2: *(a) shows a multiple tree alignment for the extended forest representation of RNA secondary structures and (b) its corresponding profile. The rows of the frequency vectors stand, from top to bottom, for the frequencies of the symbols A,C,G,U,P,λ. Note that the frequency of a base is zero iff the frequency of a base-pair bond is greater than zero.*

scores for each node. A new profile-forest can be constructed in $O(|F|)$ from an alignment of profiles using the weighted mean values of the aligned frequency vectors as the frequency vector of the profile. The number of forests in the aligned profiles determines the weight. Formally, let n and m be the number of aligned forests for the profiles P_1 and P_2, respectively. For each relabeled node in the alignment of P_1 and P_2, I define p_1 and p_2 as the aligned frequency vector in P_1 and P_2, respectively. For each insertion and deletion node, I define the p_1 and p_2 to be the frequency vector of a gap node, the vector $(0, 0, 0, 0, 0, 1)^T$. The combined frequency vector p_a is calculated as follows:

$$p_a = \frac{n}{n+m} \cdot p_1 + \frac{m}{n+m} \cdot p_2 \tag{5.7}$$

Note that also single structures in the extended forest representation can be converted to a corresponding profile.

5.3.1 Visualization of RNA Secondary Structure Profiles

A profile for a forest alignment can represent multiple RNA secondary structures and gives rise to the question to find a consensus structure. Since bases paired in one structure can be aligned to bases unpaired in another structure, this leaves some choice.

An RNA profile is *compatible* to a single structure if the leftmost and rightmost child of each P-node is a B-node. A compatible profile can be obtained from an arbitrary profile by deleting P-nodes. An optimal consensus structure corresponds to a compatible profile that maximizes the sum of P-node frequencies.

I provide a console output and a 2-d plot visualization for consensus sequence and structure.

ASCII Representation

In the ASCII representation, I draw the consensus sequence on top of the consensus structure. The height of "*" symbols on top and below the consensus sequence-structure gives the frequency of bases and base-pairs in the consensus. Each "*" means 10% frequency. Sequence and structure conservation and the relation between them can be read from this arrangement. An example is given in Figure 5.3.

2d-plot

The 2d-plot visualization for RNA secondary structures was refined to visualize consensus structures by adding reliability information to the drawing

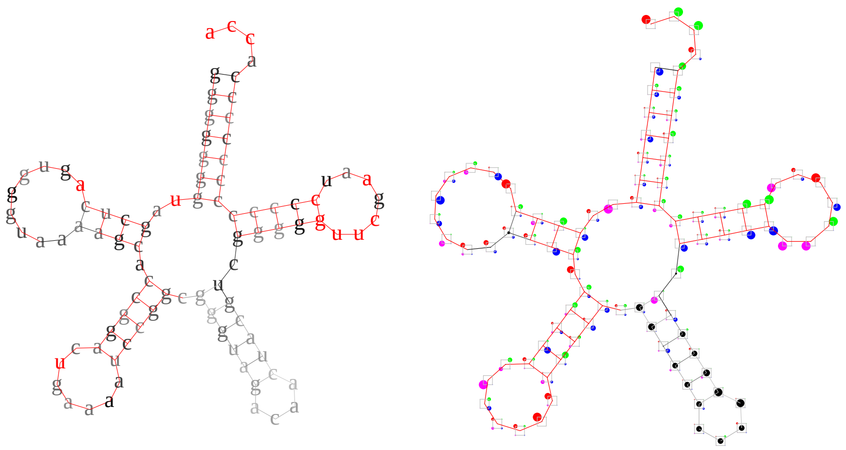
```
                  * *  ****                     ****
                  * *  ****                     ****
                 ** *  ****                     ****     *
                 ** *  ****                  *  ****     *
                 ** *  ****  ********     ****  ****     *
**************** ** *  ****  ********     ****  ****     *
**************** ** *  ****  ********     ****  ****  ****
**************** ** * ****************    ****************
**************** ** **************************************
**************** ** **************************************
ggggcuauagcucagcugggggagcuauagcucagcugggagcggggauagcuuaacc
.((((....))))....((.(.(((((..((((........))))...)))))))..))
**********************************************************
**************** ** **************************************
**************** ** ** ***********************************
**************** ** *  ***************   *********** *****
                 ** *  ****  ********    *****  **** *  **
                 ** *  ****  ********    *****  **** *  **
                 ** *  ****   *******    *** *  **** *  **
                 ** *  ****                  *  **** *  **
                  * *  ****                     **** *  *
                  * *  ****                     **** *  *
```

Figure 5.3: *ASCII representation of a consensus sequence and structure.*

using some color schemes [24, 105]. I draw the consensus structure in two forms that differ only in the way sequence information is included. Both express the frequency of base-pairs and the presence (or absence) of gaps as a gradient from light grey to black. A base-pair is only drawn if it is present in at least fifty percent[2] of the structures. With respect to sequence information, I provide either the most frequent base at each residue, or indicate the base and gap frequencies in an arrangement of colored dots[3]. In contrast to others, my visualization includes the full sequence information of the consensus structure. This information is useful to relate structure and sequence

[2] This parameter is adjustable.

[3] This arrangement was the result of a discussion with Peter Stadler who I gracefully acknowledge here.



(a) (b)

Figure 5.4: *2d-plots of a multiple alignment of 20 secondary structures of E.coli tRNAs. In both plots, the consensus structure is shown. The lighter a base-pair bond is drawn, the less frequent does it exist in the structures. Bases or base-pair bonds that have a frequency of one are drawn in red. The darkness of the lines connecting adjacent bases (the backbone, not base-pairs) is proportional to the product of frequencies that there is no gap at the residues. Again, if there is no gap the connecting line is drawn red. (a) The most frequent base at each residue is printed with the base frequency indicated by greyscale. (b) The frequencies of the bases a,c,g,u are proportional to the radius of circles that are arranged clockwise on the corners of a square, starting at the upper left corner. Additionally, these circles are colored red, green, blue, magenta for the bases a,c,g,u, respectively. The frequency of a gap is proportional to a black circle growing at the center of the square.*

conservation. Figure 5.4 shows an example of these profile drawings.

5.4 A Progressive Profile Algorithm

The alignment based comparison of RNA secondary structures allows to harness multiple sequence alignment strategies almost straightforward. Here, I introduce an algorithm that is inspired by the progressive calculation of multiple sequence alignments as in *ClustalW* [200]. In contrast to *ClustalW*, my algorithm does not calculate a guide tree solely based on initial pairwise similarities. It has been observed (A. Dress, personal communication) that any such phylogeny tends to reproduce the guide tree, no matter how well this tree really suits the data.

My strategy is as follows: As in *ClustalW*, I start with the computation of all pairwise profile distances. From these comparisons, the profiles with the highest similarities are merged and (unlike *ClustalW*) the similarity of the new combined profile to all other profiles is calculated. This procedure is repeated until only one profile is left.

A well known problem of the progressive strategy is that errors made early in an alignment cannot be rectified when further sequences are added. To reduce the greediness of my strategy, I do not simply merge the pair of profiles that obtains the highest similarity, but consider also the similarity to, and between, other profiles. Therefore, a maximum weighted matching between profiles is calculated in each step: Consider profiles as vertices in a graph. Each pair of vertices is connected by an edge that has a weight corresponding to the similarity of the profiles. A *maximum weighted matching* is a subset of edges such that no two edges share a common endpoint and the sum of edge weights is maximal. I use Gabow's N-cubed weighted matching algorithm to find the best matching pairs of profiles [51]. The pair of profiles with the highest similarity according to this matching is the one that is merged. Algorithm 5.1 computes a multiple forest alignment according to the proposed strategy.

To facilitate joining multiple pairs of profiles in Step 4, the algorithm could join the best n pairs or all pairs that exceed a certain similarity threshold. Figure 5.5 shows an example of a progressive profile alignment of RNA sec-

```
Input: Forests $F_1, F_2, \ldots, F_n$ in the extended forest representation.
Output: A profile forest $P$ for the multiple alignment of $F_1, F_2, \ldots, F_n$
1 Convert $F_1, \ldots, F_n$ into single structure profiles $P_1, \ldots, P_n$.
2 Construct all $\frac{n(n-1)}{2}$ pairwise relative similarity scores $\sigma_{\mathrm{SP\_REL}}$ of
  $P_1, \ldots, P_n$.
3 Compute a maximum weighted matching $M$ for the pairwise
  similarities.
4 Choose $P_i$ and $P_j$ of maximal similarity according to $M$, compute
  their alignment $P_{ij}$, and replace both by $P_{ij}$.
5 Compute the relative similarity score of $P_{ij}$ with all others.
6 Iterate Steps 3 to 5, until only a single profile alignment $P_{1\ldots n}$ is left.
```

Algorithm 5.1: Progressive profile alignment of forests.

ondary structures.

A related strategy has been suggested by Torsello et al. [204, 205]. In contrast to the forest alignment similarity, they compute the tree edit distance between trees. In the progressive calculation, they merge trees based on the edit distance mapping. As was shown in Section 2.5.3, such a mapping is not always consistent with a consensus tree structure. Torsello et al. simply reject the merge and search for another pair of trees to be merged.

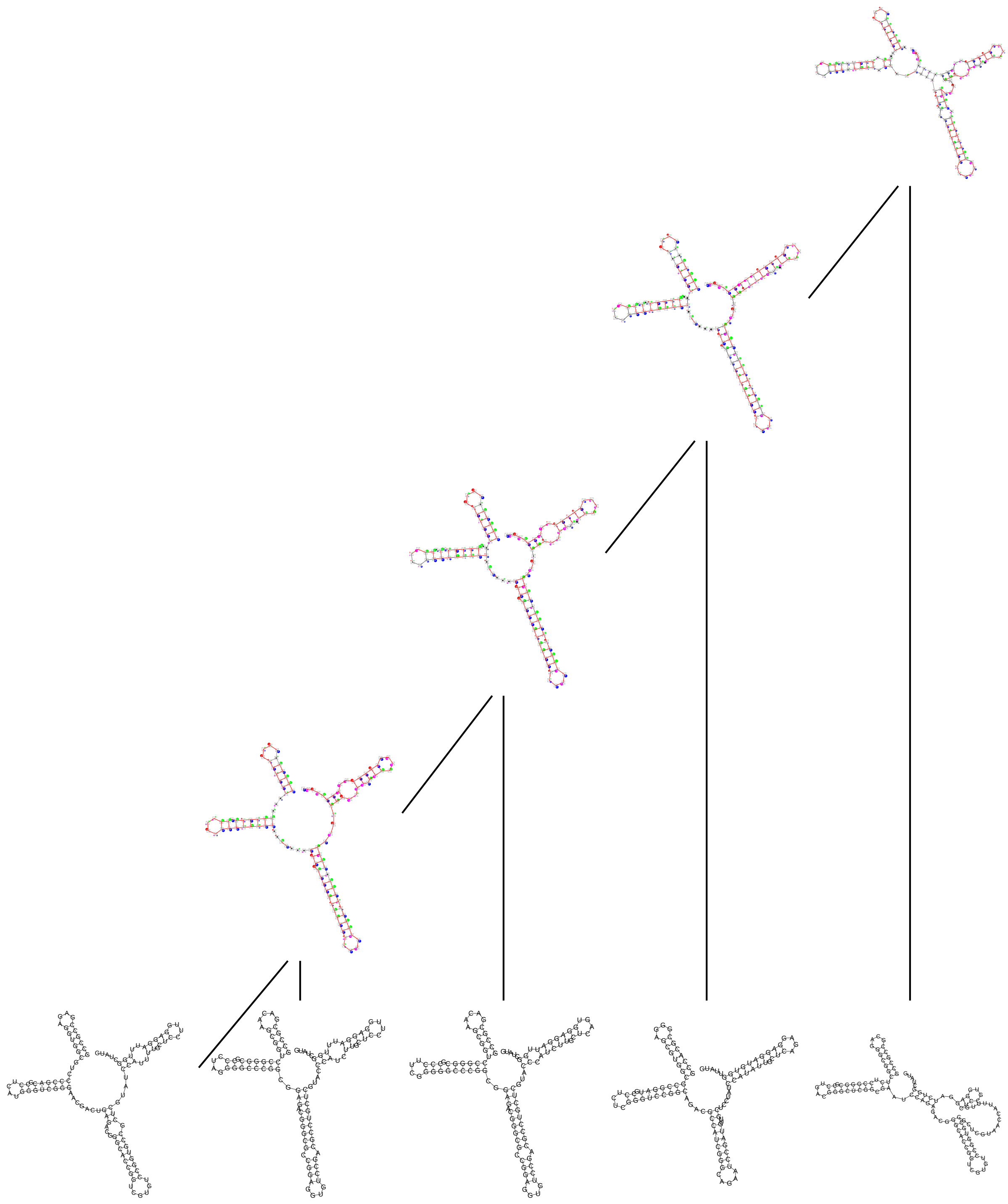


Figure 5.5: *A progressive profile alignment of predicted structures for* ROSE elements. *These genes encode for RNA molecules that have regulatory function triggered by the environmental heat, so called* RNA thermometers *[27, 146]. The structures were predicted by* RNAfold *using the default parameters. In the shown progressive alignment a single structure profile is joined to a cluster of profiles in each step. Note that this is the optimal joining procedure in this example but it is also possible to merge clusters of profiles: If the score between the two rightmost structures would be slightly better, these structures were joined and the resulting cluster would be merged with the profile of the two leftmost structures.*

5.4.1 Efficiency Analysis

Time Complexity The asymptotic time complexity of Algorithm 5.1 is as follows: Let there be n structures of average size s, measured in terms of nodes in the corresponding forest. Let d be the average degree of a forest node. The pairwise algorithm has time efficiency $O(s^2 \cdot d^2)$ and space efficiency $O(s^2 \cdot d^2)$, see Section 3.3.2. In Step 2, this algorithm is called for all pairs of tree profiles yielding time efficiency $O(s^2 \cdot d^2 \cdot n^2)$. Both, Step 3 and 4 are repeated $n-1$ times. In Step 3, a maximum weighted matching is calculated in $O(n^3)$ using Gabow's algorithm [51]. In the ith iteration, Step 4 computes $n-i$ pairwise alignment scores. Consequently, the overall runtime of Step 3 to 5 is in $O(s^2 \cdot d^2 \cdot n^2 + n^4)$. Thus, the runtime of Algorithm 5.1 is in $O(s^2 \cdot d^2 \cdot n^2 + n^4)$.

Space Complexity In Step 1, n forests are stored, requiring $O(n \cdot s)$ space. The allocated space of a pairwise alignment can be freed after the alignment score is calculated. The scores are stored in a table of size n^2. In Step 4, the optimal alignment is obtained by recalculating the alignment and a backtracking procedure in $O(s^2 \cdot d^2)$ time and $O(s^2 \cdot d)$ space. Thus, the overall space requirement of Algorithm 5.1 is $O(n \cdot s + n^2 + s^2 \cdot d^2)$.

From the observations in Section 4.8, the degree of an RNA secondary structures can be considered as a constant. Hence, multiple RNA structure alignments under the tree alignment model can be calculated with the same asymptotic efficiency as multiple sequence alignments.

5.5 A Structure Prediction Strategy based on Multiple RNA Secondary Structure Alignment

A multiple sequence alignment is often the first step in determining a consensus structure (see Section 2.4). Homologous sequence regions are matched in the alignment and force regions of sequence diversity between them to be aligned to each other. Among a family of structural RNAs, bases-pairs are expected to be less sequence conserved in an alignment than unpaired bases. Thus, regions of diversity in an alignment are subject to structural observations. It is obvious that such a strategy requires a considerable amount of sequence conservation to be successful.

I use the multiple structure alignment to predict consensus structures the other way around. First, the structure of each RNA is predicted thermodynamically. Second, a pure multiple structure alignment is computed by the algorithm presented in the previous section. A pure structure alignment means that sequence conservation is not favored by the scoring scheme (see Section 4.5). In contrast to the sequence based approach, structural conserved regions are the anchor regions of the alignment.

The success of this strategy depends largely on the accuracy of secondary structure prediction from single sequences. Unfortunately, for a set of RNA sequences that belong to the same family, the predicted structures are often diverse and not always compatible with a similar consensus structure. For instance, in the multiple alignment example in Figure 5.5 the rightmost structure does not fit “well” to the others. Looking at the suboptimal structures reveals a structure that is in better correspondence to the others. Furthermore, for a successful application of Algorithm 5.1 to the proposed structure prediction strategy the following must be guaranteed: First, all sequences share a similar structure, i.e. they belong to the same family. Second, there is only one structural conformation for the sequences.

The remedy is a clustering of predicted structures in the progressive calculation of the structural alignment. To facilitate clustering of forests, joining alignments in Step 4 is restricted to a minimal cutoff value c. If the best alignment score of P_i and P_j in Step 3 is below c, these profiles are put into the result list of clusters which are not aligned further.

The structures that are aligned are the result of structure prediction algorithms based on thermodynamics. If only the base-pair information of a prediction is used, a base is either paired or unpaired. Thus, stable and unstable base-pairs are indistinguishable. Reasonably, the deletion of a weak base-pair should not be penalized as high as the deletion of a strong base-pair. The profile representation of structures allows an elegant way to weight base-pairs by incorporating base-pair probabilities. I calculate base-pair probabilities using McCaskill's partition function algorithm and weight the P-nodes in the initial profile forests according to the base-pair probabilities. It has been observed by Gardner & Giegerich that pruning of base-pairs with a low probability can improve the results of a structural alignment of predicted structures. A threshold value p determines the minimum probability of a base-pair that is required to generate a corresponding P-node in the extended forest representation. Algorithm 5.2 shows the modified multiple structure alignment algorithm for consensus structure prediction from RNA sequences.

```
   Input: RNA sequences $S_1, S_2, \ldots, S_n$,
     clustering cutoff $c$, minimum base-pair probability $p$.
   Output: A list of profile forest.
 1 Calculate McCaskill's partition function for $S_1, S_2, \ldots, S_n$ and build
   the weighted single structure profiles $P_1, P_2, \ldots, P_n$ according to the
   mfe structure and threshold $p$.
 2 Construct all $\frac{n(n-1)}{2}$ pairwise relative similarity scores of $P_1, \ldots, P_n$.
 3 Compute a maximum weighted matching $M$ for the pairwise
   similarities.
 4 Choose $P_i$ and $P_j$ of maximal similarity according to $M$
 5 if $\sigma_{SP\_REL}(P_i, P_j) > c$ then
 6     Compute their alignment $P_{ij}$ and replace both by $P_{ij}$.
 7 else
 8     Put $P_i$ and $P_j$ in the result list.
 9 end
10 Compute the relative similarity score of $P_{ij}$ with all others.
11 Iterate Lines 3 to 8, until no profiles are left.
```

Algorithm 5.2: Progressive profile alignment of forests.

I suggest to set the clustering cutoff c to zero, as zero separates the structures that are rather similar from those that are rather dissimilar due to some similarity scoring scheme that includes positive and negative contributions.

Complexity Algorithm 5.2 calculates the partition function for all n sequences of average length s. Each prediction is done in $O(s^3)$. Thus, Step 1 requires $O(n \cdot s^3)$ time. The complexity of the remaining calculations is the same as for Algorithm 5.1. That is, the time complexity of Algorithm 5.2 is $O(n \cdot s^3 + s^2 \cdot d^2 \cdot n^2 + n^4)$ time where d is the average degree of the predicted profiles. The $O(n \cdot s + n^2 + s^2 \cdot d^2)$ space complexity remains unaffected since each calculation of the partition function requires $O(s^2)$ space.

Chapter 6

RNAforester: A Tool for Comparing RNA Secondary Structures

RNAforester is a command line based tool for comparing RNA secondary structures. It supports the computation of pairwise and multiple alignment of structures based on the models and algorithms presented in Chapter 4 and Chapter 5. The user interface follows the philosophy of the *Vienna RNA Package* [84] and will be part of the forthcoming *Vienna RNA Package Version 1.6*. The tools and technologies that are behind *RNAforester* are outlined in Section 6.1. The command line usage and options are explained in Section 6.2. The online interface of *RNAforester* is shown in Section 6.3.

6.1 Implementation Notes

RNAforester is implemented in the programming language *C++* [187]. The source code distribution is freely available at `http://bibiserv.uni-bielefeld.de/rnaforester`. The source code distribution is packaged using the *GNU Build Tools*: *autoconf* and *automake* [61]. The source code of *RNAforester* is documented using the documentation system *Doxygen* [211]. *Doxygen* can generate an on-line documentation in HTML and off-line reference manual in various formats. It can visualize the relations between the various elements by means of include dependency graphs, inheritance diagrams, and collaboration diagrams, which are all generated automatically. The documentation is extracted directly from the sources, which makes it comfortable to keep the documentation consistent with ongoing development.

The generation of 2d plots is facilitated by Milanovic &Wagner's *g2 graphics library* [136]. The clustering of profiles in the progressive calculation of multiple alignments employs Ed Rothberg's implementation of Gabow's N-cubed weighted matching algorithm [137]. The clusters that are built during the calculation of multiple alignments are written to *Graphviz* compatible files for further analysis [64].

The data structures and algorithms of *RNAforester* are designed using *C++*'s template mechanism. Templates are very useful for the implementation of generic constructs like vectors, stacks, lists, queues which can be used with any arbitrary type. Accordingly, a forest alignment can have any type of labels. In data types and algorithms the labels become a type parameter. *C++* templates provide a way to re-use source code as opposed to inheritance and composition which provide a way to re-use object code. *C++* provides two kinds of templates: class templates and function templates. In the *Standard Template Library* (STL) generic algorithms have been implemented as function templates, and the containers have been implemented as class templates. These implementations achieve excellent practical runtime, since the expensive type replacements happen at the compile time. Since I followed the STL's template philosophy, my algorithms can easily be integrated in tools beyond the scope of RNA secondary structure comparison.

6.2 User Manual

By default, *RNAforester* calculates pairwise similarity between RNA secondary structures under the scoring scheme proposed in Section 4.5.1. *RNAforester* reads RNA secondary structures from *stdin* in *Fasta* format where matching brackets symbolize base-pairs and unpaired bases are represented by dots. An example is given below:

```
> test
accaguuacccauucgggaaccggu
.((..(((...)))..((..)))).
```

All characters after a "blank" are ignored and all "-" characters are removed. The program will continue to read new structures until it encounters a "@" character or the end of file. Lines starting with ">" can contain a structure name.

The similarity scores, alignments, and consensus sequences and structure are written to *stdout*. The default format for alignments is *ClustalW* format.

6.2.1 Options

RNAforester has a number of options that control the alignment mode and the output. In the following description, *int* and *dbl* stand for integers and floating numbers, respectively.

--help: Shows the synopsis of *RNAforester*.

--version: Shows version information of *RNAforester*.

-f=filename: This option lets *RNAforester* read input from *filename*. 2d-plots are written to files prefixed with *filename*.

-d: This option lets *RNAforester* calculate distance instead of similarity. In contrast to similarity, scoring contributions are minimized. This parameter cannot be used in conjunction with multiple alignment. (This restriction is due to technical difficulties with the maximum weighted matching algorithm used in the progressive calculation of multiple alignments.)

-r: Calculate relative similarity scores for pairwise alignments, see Equation (5.6) in Section 5.3.

-l, -s, -so=int: Option *-l* and *-s* let *RNAforester* calculate local similarity and small-in-large similarity (see Section 4.6). If parameter *-so* is used additionally, suboptimal solutions are calculated such that the score is within *so* percent of the optimum.

-m, -mc=dbl, -mt=dbl: Option *-m* activates the multiple alignment mode of *RNAforester*. The clustering cutoff can be adjusted by parameter *-mc*, the default value is zero. To facilitate joining multiple clusters in each step, parameter *-mt* can be adjusted (see Section 5.5). The default value is 0.7 (relative similarity). All pairs above this threshold are joined in each step of the multiple alignment calculation (see Section 5.4). The clusters are written to a file `cluster.dot` in *Graphviz*'s dot format. If a filename was specified by parameter *-f* the filename is "filename_cluster.dot".

-p, -pmin=dbl: Structures are predicted from the partition function algorithm in the *Vienna RNA library*. The P-nodes (representing base-pair bonds) in the corresponding forests are weighted according to base-pair probabilities from the partition function (see Section 5.5). Parameter *-pmin* sets the minimum frequency that is required to create a P-node in the extended forest representation. The default value is 0.5. By this parameter, a pruning of high entropy base-pairs is possible.

-cmin=dbl: This parameter sets the minimum frequency that is required for a base-pair to appear in final the consensus structure.

-pm=int, -pd=int, -bm=int, -br=int, -bd=int: Set the scoring values for a base-pair bond match, a base-pair bond deletion, a base match, a base replacement, and a base deletion according to the scoring model described in Section 4.5.1. The default values are *-pm*=10, *-pd*=-5, *-bm*=1, *-br*=0, and *-bd*=-10.

--RIBOSUM: Uses the scoring model described in Section 4.5.2 with the RIBOSUM85-60 matrix. The RIBOSUM score is only supported for pairwise alignments.

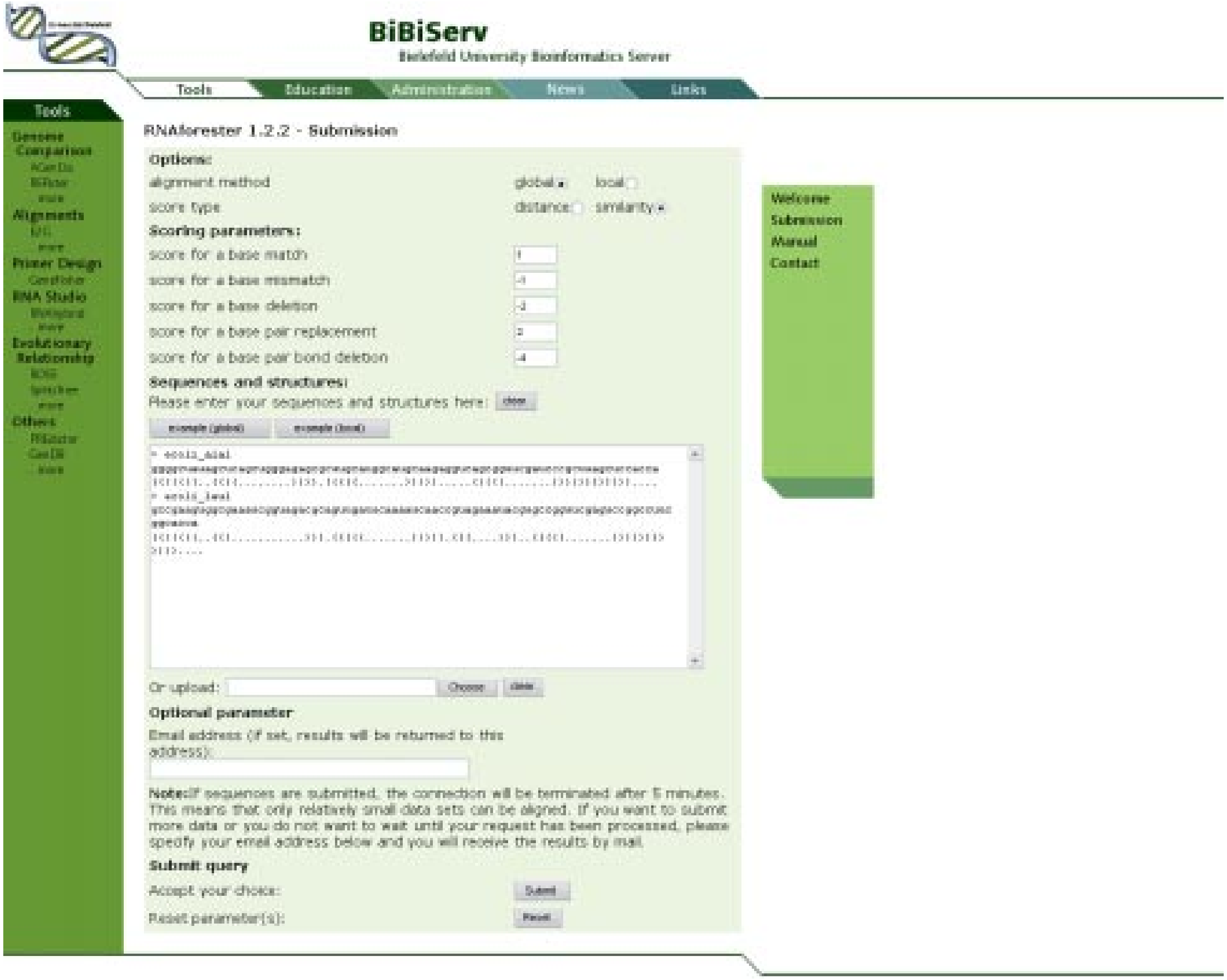
-2d, --2d_hidebasenum, --2d_basenuminterval=int, --2d_grey, --2d_scale=dbl, --2d_png, --2d_jpg: Option *-2d* activates the generation of 2d-plot postscript files. In the pairwise alignment mode, the drawings are written to files `x_n.ps` and `y_n.ps` where *n* is an index. If local similarity (*-l*,*-s*) in conjunction with suboptimal solution (*-so*) is set, *n* enumerates the suboptimal solutions. The region of local similarity are highlighted in the 2d-plots of the original structures that are written to the files `x_str.ps` and `y_str.ps`. Parameter *--2d_hidebasenum* disables the numbering of bases according to the interval *--2d_basenuminterval*. Colors can be turned into gray-scale by parameter *--2d_grey*. The size of the drawing can be adjusted by parameter *--2d_scale*. The drawing is scaled by the given factor. The parameters *--2d_png* and *--2d_jpg* let *RNAforester* write 2d-plots in *PNG* and *JPG* format.

--fasta: Alignments are printed to the console in *Fasta* format.

--score: Only the optimal score of an alignment is printed. This option is useful when *RNAforester* is called by another program that only needs a similarity or distance value.

6.3 RNAforester Web Interface

The online version of *RNAforester* is available at the *Bielefeld Bioinformatics Server* (`http://bibiserv.uni-bielefeld.de/rnaforester`) [176]. A screenshot is shown in Figure 6.1. For more informations refer to the online manual.



RNAforester 1.2.2 - Submission

Options:

alignment method global ● local ○

score type distance ○ similarity ●

Scoring parameters:

score for a base match 1

score for a base mismatch -1

score for a base deletion -2

score for a base pair replacement 2

score for a base pair bond deletion -4

Sequences and structures:

Please enter your sequences and structures here:

```
> [illegible]
[illegible]
[illegible]
> [illegible]
[illegible]
[illegible]
[illegible]
```

Or upload:

Optional parameter

Email address (if set, results will be returned to this address):

Note: If sequences are submitted, the connection will be terminated after 5 minutes. This means that only relatively small data sets can be aligned. If you want to submit more data or you do not want to wait until your request has been processed, please specify your email address below and you will receive the results by mail.

Submit query

Accept your choice:

Reset parameter(s):

Figure 6.1: *Screenshot of the web interface of* RNAforester*.*

Chapter 7

Applications

In this chapter, I demonstrate the practical impact of the Algorithms that were presented in this thesis. In Section 7.1 I present a joint work with T. Töller and R. Giegerich that is initially described by Töller in [203]. We present a pipeline for the detection of new regulatory motifs that, as an integral part, includes the computation of local structure alignments. Multiple alignment of RNA secondary structures is helpful to reveal a common structural property of RNAs. We present two applications for multiple structure alignment, motif discovery and consensus structure prediction in Section 7.2. The latter application is initially based on sequence information where no verified structures are given as proposed in Section 5.5.

7.1 Local Structure Alignment as a New Strategy for RNA Motif Detection

This section presents joint work with T.Töller and R. Giegerich. The investigation of structural RNAs or RNA motifs is a major task in modern molecular biology. Untranslated terminal regions (UTRs) of mRNAs sometimes contain regulatory motifs which are important for the posttranscriptional gene regulation. Such motifs can affect mRNA localization [91], mRNA degradation [69], and translational regulation [65]. One of the best investigated regulatory motifs in UTRs is the *Iron Responsive Element* (IRE). It is a specific stemloop structure that can be found in the 5'UTRs and 3'UTRs of various mRNAs [101]. There is for example one IRE in the 5'UTR of the vertebrate ferritin mRNAs where it regulates the translational efficiency depending on the amount of iron in the cell. If there is no iron in the cell regulatory proteins bind to the IRE which results in a translational block of the ferritin mRNA. In contrast, protein binding to five IREs in the 3'UTR of the human transferrin receptor mRNA leads to a stabilization of this mRNA at a low iron-level in the cell. Thus, the same structural RNA motif functions in different posttranscriptional regulatory pathways depending on its location in the 5'UTR or 3'UTR.

7.1.1 Strategies for the Detection of RNA Motifs

Regulatory RNA motifs like the IRE often consist of both, sequence and structure features. Therefore special requirements exist for the prediction of such regulatory RNA motifs. There are essentially three strategies that can be used for RNA motif detection (refer also to Section 2.4).

The *simultaneous strategy* is the joint optimization of sequence alignment and RNA folding and was first postulated by Sankoff [172]. But because of its time complexity it cannot be used for real data.

The *sequenced based strategy*, in the initial step, calculates a sequence alignments to identify regulatory motifs by their conservation on sequence level. In a second step an RNA folding program like *RNAfold* can be used to verify that the conserved sequences can built a common structure motif. Since regulatory motifs in RNAs are often more conserved in structure then in sequence this strategy will fail to identify such motifs.

The *pure structure alignment strategy* (for short *pure strategy*) was introduced and applied by Töller in [203]. It is based on RNA folding and subsequent detection of conserved motifs. These are purely structure motifs and require no sequence alignment - this explains the name of the strategy. The calculation of local structure alignments with the algorithms that were introduced throughout this chapter, implemented in the tool *RNAforester* (see Chapter 6), is an essential step of the strategy. The *pure strategy* follows the protocol shown in Figure 7.1, and will be explained throughout the next sections. We will report on a viability study using IRE motifs, and on the prediction of a new regulatory motif and its wet lab validation.

7.1.2 The Pure Structure Alignment Strategy

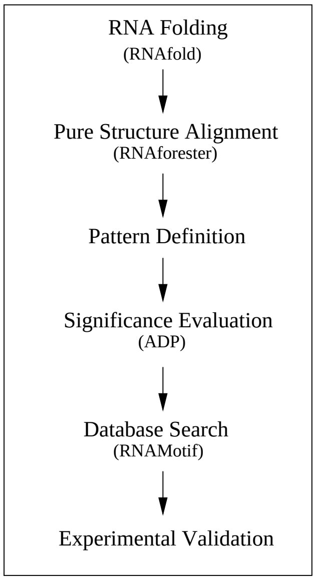


Figure 7.1: *Essential Steps of the Pure Structure Alignment Strategy.*

The *pure strategy* (see Figure 7.1) comprises six steps where RNA folding, structure alignment, significance evaluation and pattern search are based on suitable Bioinformatics methods. All steps may include some variation of parameters. Pattern design and significance analysis is a somewhat mathematical activity, while validation means experimental work with its typical fallacies. We describe these steps and the considerations that guide them in the context of two applications of our strategy.

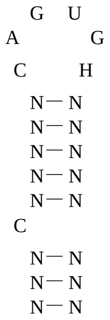


Figure 7.2: *Iron Responsive Element: eukaryotic consensus structure (H: A,C,U). The cytosin bulge can be extended to an internal loop.*

Proof of Concept

To validate the *pure strategy* we focus on the investigation of structural motifs in untranslated terminal regions (UTRs) of mRNAs. As mentioned before one of the best investigated regulatory motifs in UTRs is the *Iron Responsive Element*. Figure 7.2 shows a consensus structure of eukaryotic IREs. It is a specific stem-loop structure which consists of a helix region that contains a cytosin bulge (this bulge is sometimes extended to an internal loop) and a loop of six bases with a consensus sequence. All this knowledge is not to be used in our proof-of-concept study.

UTRs which are known to contain IREs were taken from the UTR data base [154]. We chose the ferritin 5'UTR from human and the succinate dehydrogenase 5'UTR from *Drosophila* which are in a size range of 200 nucleotides and both contain one IRE. The *pure strategy* was also applied for the detection of an IRE in a larger UTR, namely the transferrin receptor 3'UTR which consists of nearly 2500 nucleotides and contains five IREs. First the mfe (minimal free energy) structures of the UTRs were predicted. Because we are interested in regulatory motifs which can be seen as small substructures in the complete UTR secondary structures, we calculated local structure alignments with *RNAforester*.

In Figure 7.3 the local alignment of the 5'UTRs of the human ferritin heavy chain mRNA and the Drosophila succinate dehydrogenase mRNA is

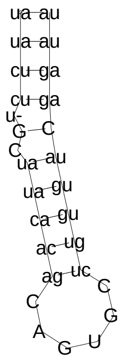


Figure 7.3: *Local structural alignment of the 5'UTRs of the human ferritin heavy chain mRNA and the drosophila succinate dehydrogenase mRNA, extracted as the best common motif from two structures comprising about 200 bases. The stem regions of both IREs differ extremely in sequence.*

displayed. The IRE was detected as the most similar substructure in both UTR secondary structures.

Because it is not guaranteed that the energetically best structure is the biologically correct one, suboptimal structures should always be investigated too. In this example we were successful by aligning only the two mfe structures, which contained the IRE. The investigation of larger structures like the transferrin receptor 3'UTR, structure prediction becomes a general problem. Structure predictions based on thermodynamic parameters are only reliable for smaller structures and even then, energetically suboptimal structures have to be considered. Still, folding long sequences and investigation of the structures does make sense for finding smaller motifs in these larger structures. If a structural RNA motif has an important biological function (e.g. the IRE) it should be very stable and we can expect to find it in the structure prediction of a long sequence even if the folding of the complete sequence makes no sense biologically. To show this we have calculated the local structure alignment of the human ferritin heavy chain 5'UTR (208 nucleotides) and the human transferrin receptor 3'UTR (2464 nucleotides). We detect the IRE again (see Figure 7.4), although it occurs at completely different positions in the two UTRs. Thus, using the *pure strategy* for RNA motif detection we are not

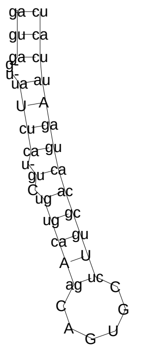


Figure 7.4: *Local structural alignment of the 5'UTR of the human ferritin heavy chain mRNA (208 nucleotides) and the 3'UTR of the human transferrin receptor mRNA (2464 nucleotides). The IRE was detected as the most similar motif in both structures.*

restricted to small structures.

It is important to note that we are able to discover regulatory motifs solely by their structural preservation, and independent of their sequence conservation and position in the UTR. The further steps of the *pure strategy* according to Figure 7.1 will be presented in the next section, where we describe the prediction of a potential new regulatory RNA motif.

Prediction of a new Regulatory RNA Motif in the RAB1A 3'UTR

After demonstrating the viability of the *pure strategy* using the familiar IRE motif, we ventured out to discover something new.

RAB1A is a ubiquitous protein with a role in Endoplasmatic Retikulum (ER) to Golgi transport [128]. In previous work a high sequence conservation in several vertebrate RAB1A 3'UTRs was shown [228]. Therefore a function of a structural motif for the posttranscriptional regulation of the RAB1A mRNA is possible. The *pure strategy* was used for the prediction of potential regulatory elements in the RAB1A 3'UTR. We started with the investigation of the human and electric ray RAB1A 3'UTRs, which have much less sequence-conservation than the UTRs described in [228]. In Figure 7.5 the mfe structure of a part of the human RAB1A 3'UTR and in Figure 7.6 the

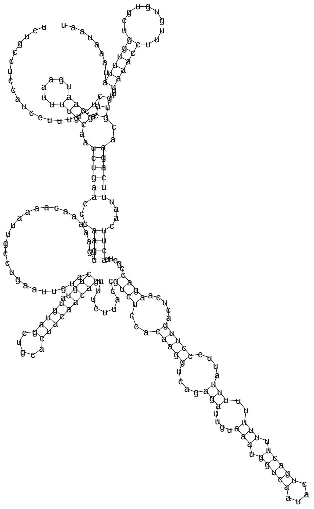


Figure 7.5: *mfe structure for a part of the human RAB1A 3'UTR.*

mfe structure of the electric ray RAB1 3'UTR is displayed.

For these structures a local alignment was calculated using *RNAforester*. Figure 7.7 shows this alignment. The detected stemloop is not only highly conserved in structure but also in sequence. Although *RNAforester* can make use of such sequence similarity, the scoring contribution for a base-matches was set to zero[1], thus purely relying on structure conservation. Analysis of base pair probabilities confirmed the stability of this stemloop in many energetically suboptimal structures (data not shown).

For the computational validation of the predicted stemloop we performed a database search. First a search pattern was defined that should be general enough to find as many occurrences as possible. At the same time it should be specific enough to find as few false positives as possible. Therefore we used significance evaluation based on the ADP method [58, 135] for the definition of a search pattern. Figure 7.8 shows the pattern for the predicted stemloop.

[1] A small contribution of sequence similarity in comparison with structural similarity is also feasible. Loop regions would be aligned on the sequence level without dominating the sequence structure alignment, see Section 4.5)

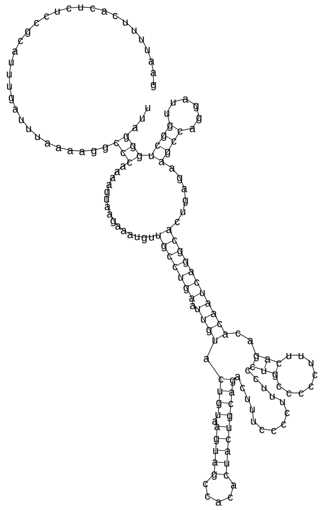


Figure 7.6: *mfe structure for the electric ray RAB1A 3'UTR.*

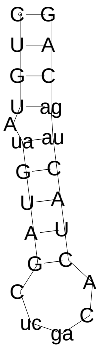


Figure 7.7: *Local structure alignment of the RAB1A 3'UTRs from human and electric ray. The resulting stemloop is also highly conserved in sequence with few compensatory base exchanges in the helix.*

```
        C           A
     Y    GNNNN  NNNN-5'
   R      |||||  ||||
     C    CNNNN  NNNN-3'
        A
```

Figure 7.8: *RAB1 stemloop pattern for the database searches. The adenine bulge and the loop with the closing base pair was fixed in sequence with some variability in the second (U or C) and third (A or G) position.*

The length of the helix was fixed but the sequence was kept variable. Bulges promote RNA-protein interactions, thus the adenine bulge is an important element of the pattern. The loop sequence was predefined with partial variations at the second and third position. Pattern design and significance evaluation is an iterated process. For the resulting pattern, we computed an expectation value of 0.8 hits in a random sequence with size and base composition of the 3'UTR collections of the UTRdb version 15.0 [154]. These collections contain about 47 million bases. For the database search we used the program *RNAMotif*. Combined sequence and structure motifs can easily be described within a descriptor file and that file can than be used for database searches. Our defined pattern for the RAB1 stemloop was described with *RNAMotif* as follows:

```
parms
    wc += gu;

descr
  h5(tag='stem1', len=4)
     ss(len=1, seq="a")
     h5(tag='stem2',len=5,seq="g$")
       ss(len=5, seq="cyrca")
     h3(tag='stem2', seq="^c")
  h3(tag='stem1')
```

The 5' site of the first helix with fixed length 4 is followed by the adenin bulge and the 5' site of the second helix with fixed lenght 5 and a guanine at the end. The loop region of length 5 has variable sequence at position 2 (uracil or cytosin) and position 3 (guanine or adenine). The 3' site of the second helix must start with a cytosin. We used this pattern for searching the 3'UTR collections of the UTRdb. The result of this search is summarized in Table 7.1.

Significantly more hits (13) than would have been expected in a random database (0.8) were found and these hits were exclusively to RAB1 and Sir2α 3'UTRs. There were no hits found in the 3'UTRs of invertebrates or viruses

UTRdb collection	**Hits**
Human 3'UTR	RAB1A Sir2α
Rodent 3'UTR	RAB1A (mouse) Sir2α, clone (similar to Sir2α)(mouse)
Other Mammals 3'UTR	RAB1A (cat), RAB1A (opossum), RAB1A (bull), RAB1A (quolls), RAB1A (kangaroo)
Other Vertebrates 3'UTR	RAB1A (alligator), RAB1A (chicken) RAB1 (electric ray)
Invertebrates 3'UTR	——
Virus 3'UTR	——

Table 7.1: *Results of the database search for the RAB1A stemloop pattern. The stemloop occured only in RAB1A and Sir2α 3'UTRs of vertebrates. There were no hits in 5'UTRs.*

or in any of the 5'UTR collections. This makes a biological function of the stemloop very likely.

Gelmobility-Shift Experiments

To get further hints for the biological function of the stemloop we did several laboratory experiments. Posttranscriptional gene regulation is often the result of specific protein interactions with regulatory motifs in UTRs. Therefore protein binding to the predicted stemloop is very likely if the stemloop has a biological function. We performed gelmobility-shift assays for showing such an RNA protein interaction (see Figure 7.9).

Digoxigenin labeled RNA oligos whose sequence matched the mouse (and also human) stemloop (Lane 1) were incubated with protein extract from mouse kidney (Lanes 2-5). In Lanes 3-5 complex formation was reduced using rising amounts of unlabeled RNA oligos. The control in Lane 6 (only protein extract) shows that the band on the top is the result of a cross reaction of the Digoxigenin antibody with the protein extract. Below this band a small complex band can be seen but most of the labeled oligos didn't run into the gel and we assume that's the result of a large protein complex interacting with the RNA stemloop. We also tried to isolate the binding proteins using RNA oligos bound to magnetic beads and here we found lots of protein bands (data not shown) which can be another hint for a large protein

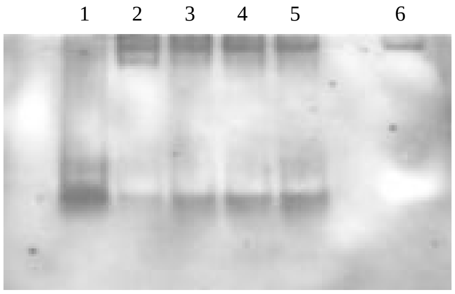


Figure 7.9: *Gelmobility-shift assay: lane 1: RNA-Oligo (DIG-labeled), lane 2: RNA-Oligo(DIG) + protein extract, lane 3: RNA-Oligo (labeled:unlabeled= 1:50) + protein., lane 4: RNA-Oligo (labeled:unlabeled = 1:150) + protein, lane 5: RNA-Oligo (labeled:unlabeled= 1:300) + protein, lane 6: only protein as negative control for DIG-detection; lanes 2-5 contained an excess of yeast RNA as an unspecific competitor.*

complex interacting with the stemloop (even though some unspecific RNA protein interactions might occur). Although more experiments have to be done to elucidate the exact biological function of the predicted stemloop, the high conservation of the stemloop in different vertebrates, its main restriction to RAB1 3'UTRs and the first experimental hints for specific protein interactions with the stemloop let us assume, that we found a new RNA motif for posttranscriptional gene regulation.

The new Potential Regulatory Motif in the RAB1A 3'UTR The *pure strategy* was used for the prediction of a structural motif in the RAB1A 3'UTRs of human and electric ray. The predicted stemloop is very stable and highly conserved in sequence. Although we created a search pattern for this stemloop, that is highly variable on sequence level in the stem region, a database search in the UTRdb showed hits only in RAB1A 3'UTRs of 10 vertebrates and also in the Sirtuin 2α 3'UTR of human and mouse. This restriction of the stemloop to only 2 different mRNAs makes a biological function very likely. The gelmobility-shift assays revealed protein interactions with the stemloop and in ongoing experiments we try to identify the binding proteins for getting more information about a possible posttranscrip-

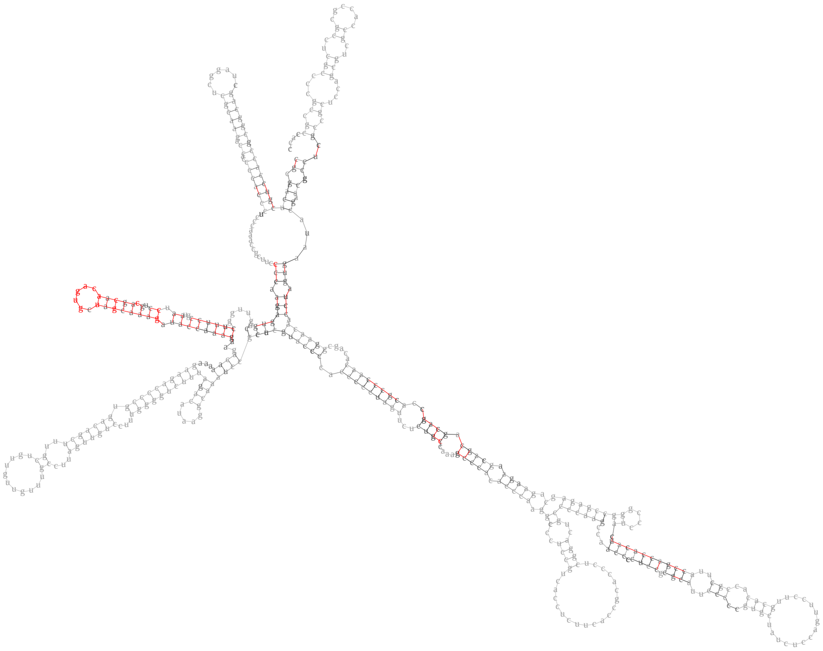


Figure 7.10: *Multiple alignment of the four 5'UTRs of human and mouse ferritin heavy chain mRNA (5HSA015337, 5MMU002159) and SLC11A3 iron-transporter mRNA (5HSA023193, 5MMU011005). The alignment clearly superposes the IRE elements, automatically marked red by our visualization*

tional gene regulation of the RAB1A mRNA. Because the RAB1A protein is localized near the *cis*-golgi membrane [174] we assume that the RAB1A mRNA could be localized previously. Identification of the binding proteins should give us hints for a role of the stemloop in a possible localization of the RAB1A mRNA.

7.2 Multiple Alignment

7.2.1 Motif Discovery

Multiple structure alignment can be used for searching regulatory structural motifs common to several RNAs. One of the best investigated regulatory motifs is the iron responsive element (IRE), which is a specific stem-loop structure and can be found in the untranslated terminal regions (UTRs) of many mRNAs. It regulates for example the translational efficiency of these mRNAs according to the amount of iron in the cell [10]. The 5'UTRs of human and mouse ferritin heavy chain mRNA and SLC11A3 iron-transporter mRNA were taken from the UTR data base [154]. These UTRs are known to contain iron responsive elements. Their secondary structures were predicted with *mfold* (Version 3.1) [244] and a multiple structure alignment of the UTRs was calculated using *RNAforester*. In Figure 7.10, the resulting alignment is displayed. The red colored stemloop shows the conserved iron responsive element that occurs in all structures. All other structural elements shown in black or gray can only be found in some of the structures. Thus, the described approach is useful for the detection of common structural motifs in a set of RNA secondary structures. This example works well because the element of interest resides in similar positions in the globally aligned structures. Should this positions vary, a local similarity comparison can be employed [77]. Unfortunately, this is restricted to pairwise comparisons.

7.2.2 Consensus Structure Prediction

In this Section, I exemplify the structure prediction strategy proposed in Section 5.5 that is based on a multiple structure alignment of thermodynamically predicted structures. Throughout this section this strategy is referred to as *the structure alignment strategy*. The converse to the the structure alignment strategy is a strategy that first calculated a multiple sequence alignment and then derives a consensus structure by analyzing covariance and thermodynamic considerations. This strategy is referred to as *the sequence alignment strategy*.

Structure prediction strategies that build upon an initial multiple sequence alignment are limited in their success if the sequence identity is too high or too low. In the first case, the covariance of conserved base-pairs is low and the prediction is guided mainly by thermodynamics. In the second case, the quality of the sequence alignment is often too low in a biological sense and, hence, covariance can not be inferred from the multiple alignment. In particular, the objective function for a multiple sequence alignment aims for maximization of identity and penalizes covariance. According to McCutcheon & Eddy, for multiple sequence alignment based strategies, "the 'sweet spot' is at $\cong 75 - 85\%$ sequence identity" [134]. Washietl & Hofacker gave a slightly lower bound stating "we can conclude that there is obviously no need for structure alignments above 65% pairwise identity [223]. Thus, a good candidate family for exemplifying my strategy should have lower sequence homology than 70% to demonstrate that the structure alignment strategy is suitable to predict a common fold. The structure alignment strategy depends on predicted structures from single sequences and the prediction accuracy gets the worse the longer the sequences are. From personal experience, the sequence length should be less than 300.

The RNA families for my experiments are taken from the *Rfam* database (Version 6.1, August 2004) [67, 68]. *Rfam* is a large collection of multiple sequence alignments and covariance models covering many common non-coding RNA families. The covariance models in *Rfam* result from hand-crafted multiple sequence alignments that were collected from serious publications. These alignments are the *seed alignments* in the *Rfam* database. From several interesting candidates, I choose two families of riboswitches, the *Lysine Riboswitch* and the *TPP Riboswitch*, and a family of splicosomal RNA, the *U1 spliceosomal RNA*.

For the following experiments, I used *RNAforester* for the structure alignment strategy. Note that the prediction of structures is done automatically by *RNAforester* as proposed in Section 5.5 where the base-pairs of the prediction are weighted according to the base-pair probabilities. I use the pure structure scoring scheme proposed in Section 4.5. The clustering threshold c is zero. According to the observations of Gardner & Giegerich, pruning high entropy base-pairs can improves the results of structural comparison for predicted structures [55]. Therefore, I set the minimum probability p that is required for base-pairs to occur in the predicted structures to 0.8. The cluster join threshold t is set to 0.7. Except for the minimum probability p these are the standard setting for *RNAforester*. The command line for *RNAforester* for these settings is: `RNAforester -p -2d -pmin=0.8 -f=sequences.fas` where `sequences.fas` is the file containing the RNA sequences in *Fasta* format. For the sequence alignment strategy I calculate multiple sequence alignment using the online Version of *ClustalW* from the *European Bioinformatics Institute* [23]. I use the default parameters. The structure prediction form the multiple alignment is done by *RNAalifold* again using default parameters [82]. The score of an *RNAalifold* prediction consists of an energy term (first term) and a covariance term (second term). Recently Washietl & Hofacker provided a method how to test a multiple sequence alignment for the existence of an unusually stable prediction. Their method relates *RNAalifold* predictions of a given multiple sequence alignment to the predictions of shuffled alignments. The significance is assessed in terms of *z-scores*[2]. In their experiments, a Z-score below -3 have a false positive rate below 1%. For the calculation of Z-scores, I used the *Perl* program *alifoldz.pl* as provided in the supplemental material of [223]. *alifoldz.pl* computes two scores, one for the forward and one for the backward strand of the sequences. I did no further fine tuning of parameters for any of the tools used for the following experiments.

Lysine Riboswitch

Riboswitches are metabolite binding domains within certain messenger RNAs that serve as precision sensors for their corresponding targets. Allosteric rearrangement of mRNA structure is mediated by ligand binding, and this results in modulation of gene expression. This family includes riboswitches

[2] A Z-score is a measure of the distance from the mean of a distribution normalized by the standard deviation of the distribution. Mathematically: Z-score = (value-mean)/standard deviation. Z-scores are useful for quantifying how different from normal a recorded value is. Z-scores are particularly useful when combining or comparing different features or measures. A Z score of 0 represents the mean of counts for all periods. Assuming a normal distribution, Z scores of -1, -2, -3 and +1, +2, +3 indicate that about 67%, 95% and 99%, respectively, of all values are expected by change to fall within this count. In short, higher (in absolute value) Z scores are likely to be more statistically significant in their deviation from the mean.

that sense lysine in a number of genes involved in lysine metabolism [126].

The 48 sequences from the *Rfam* seed alignment for the *Lysine Riboswitch* (Accession number: RF00168) have an average length of 181.3 and an average identity of 48%. The published consensus structure is shown in Figure 7.11. *RNAforester* outputs six clusters that contain more than one structure. The consensus structure drawings for these clusters are shown Figure 7.12-7.17. The structure in Figure 7.12 is in good correspondence with the published one. The clusters in Figure 7.13-7.17 share at most smaller regions with the published structure. Apparently, The relative sum-of-pairs $\sigma_{\mathrm{SP\_REL}}$ score for the clusters does not correlate with the reliability of the predictions. However, looking at the sequence level, the consensus structure in Figure 7.12 have a considerable amount of sequence variation while the others are highly sequence conserved. I identify correct predictions based on the following hypothesis: *The more structurally conserved and the less sequence conserved a multiple alignment is, the more reliable are the predicted structures.* In contrast to the sequence alignment strategy that uses covariation to predict structures, in the structure alignment method thermodynamic predictions are validated by covariance. So far, I only consider sequence identity to identify the best cluster.

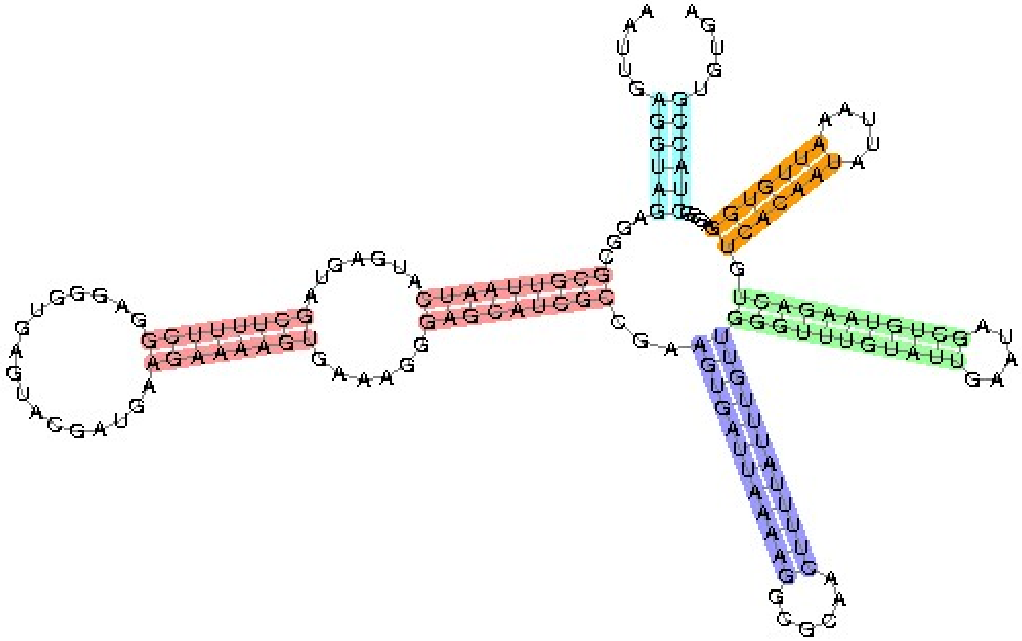


Figure 7.11: *Consensus structure of the Lysine riboswitch as published in [126].*

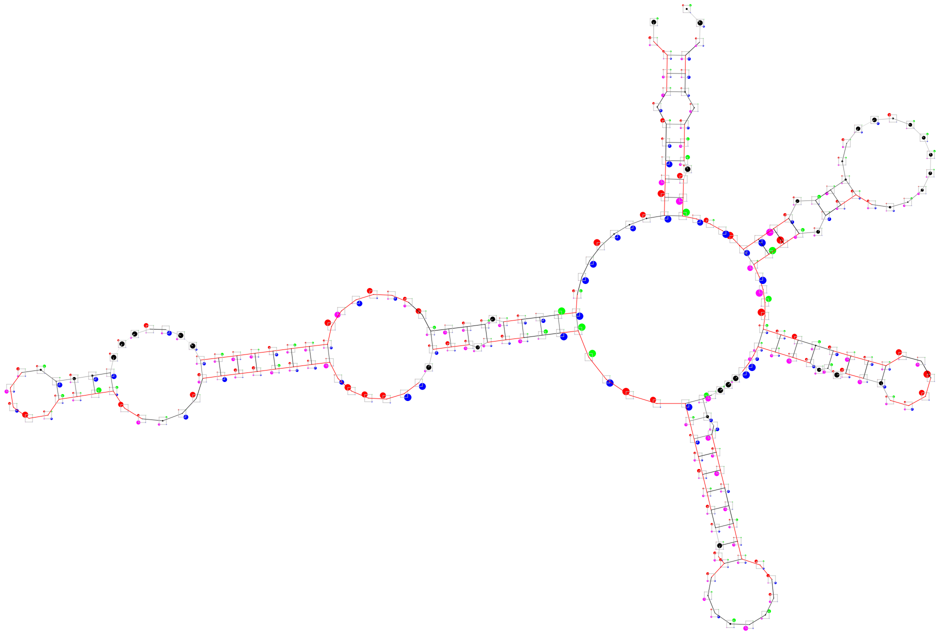


Figure 7.12: *Lysine Riboswitch. Consensus structure of 18 sequences as predicted by RNAforester. The sum-of-pairs score σ_{SP} for this cluster is* 436.177.

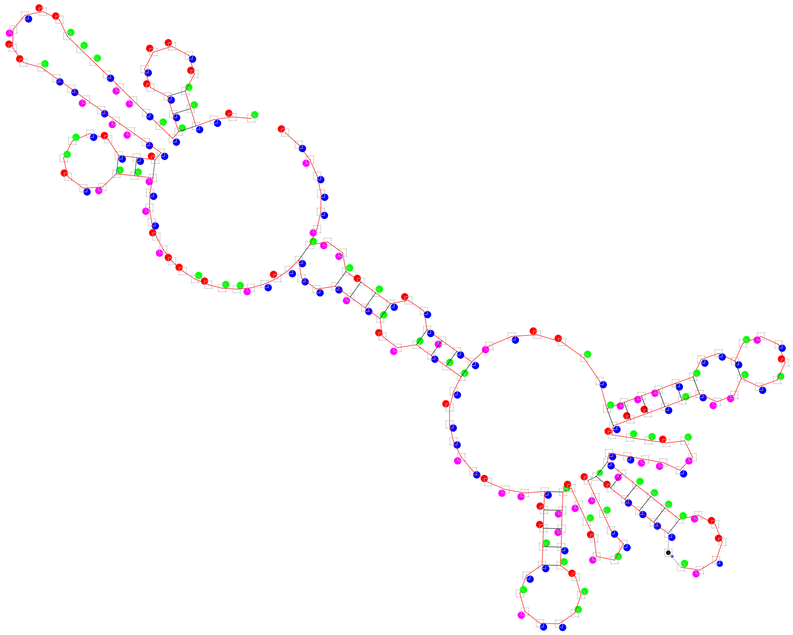


Figure 7.13: *Lysine Riboswitch. Consensus structure of 7 sequences as predicted by RNAforester. The sum-of-pairs score σ_{SP} for this cluster is* 485.696*.*

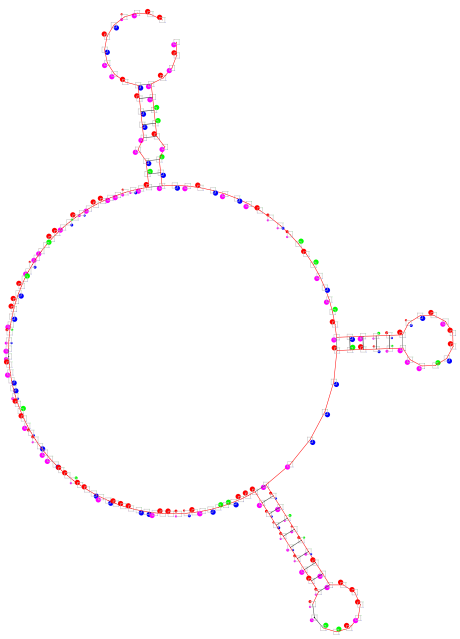


Figure 7.14: *Lysine Riboswitch. Consensus structure of 7 sequences as predicted by RNAforester. The sum-of-pairs score σ_{SP} for this cluster is* 312.722

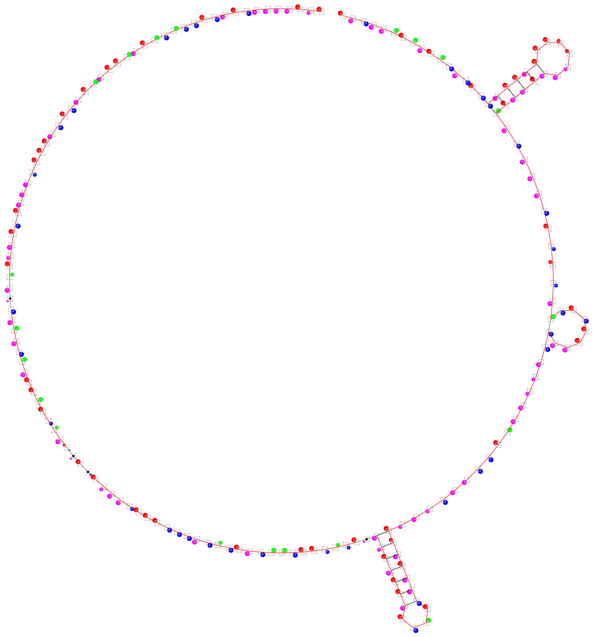


Figure 7.15: *Lysine Riboswitch. Consensus structure of 5 sequences as predicted by RNAforester. The sum-of-pairs score σ_{SP} for this cluster is* 349.197

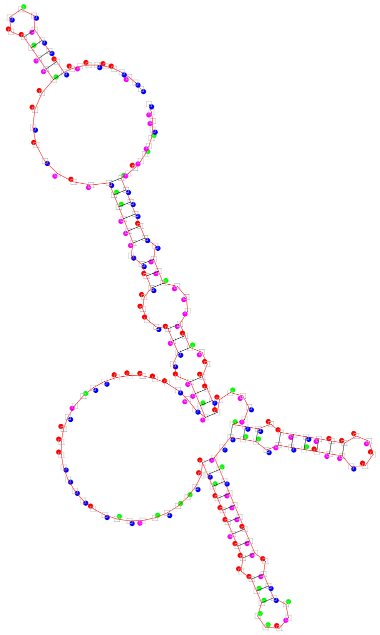


Figure 7.16: *Lysine Riboswitch. Consensus structure of 3 sequences as predicted by RNAforester. The sum-of-pairs score σ_{SP} for this cluster is* 562.263

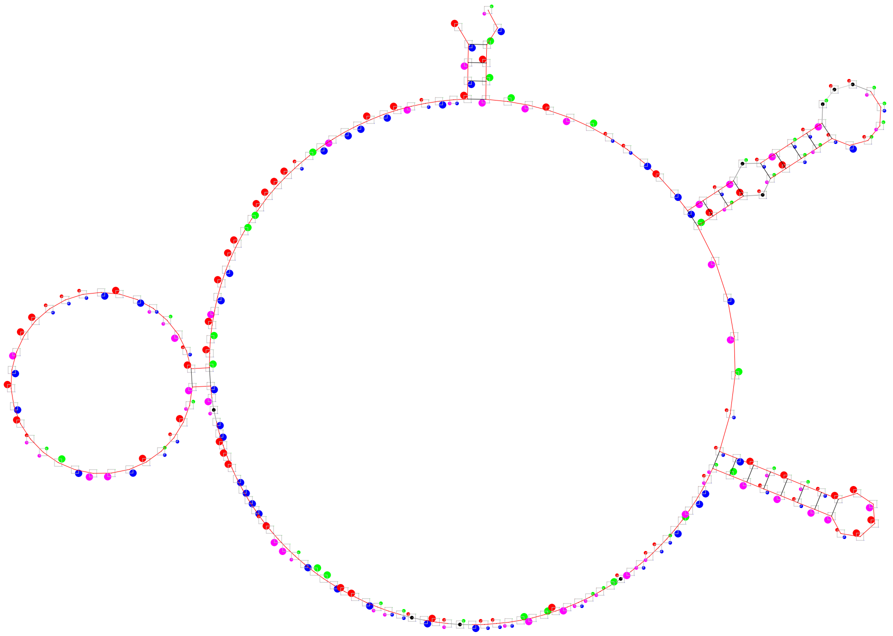


Figure 7.17: *Lysine Riboswitch. Consensus structure of 2 sequences as predicted by RNAforester. The sum-of-pairs score σ_{SP} for this cluster is* 414.111

A *RNAforester* structure alignment produces a sequence alignment as a coproduct[3]. In the following, I compare the results of the structure alignment strategy to results of the sequence alignment strategy. Figure 7.18 shows the *RNAalifold* prediction for the hand-crafted seed alignment from the *Rfam* database. This prediction is in good correspondence with the published one. Figure 7.19 shows the prediction for the *ClustalW* alignment of the seed sequences. Clearly, the sequence alignment can not arrange the bases such that *RNAalifold* can derive a common structure. Since *RNAforester* does a clustering of the structures, I also compare the *RNAalifold* prediction for sequence alignment derived from *RNAforester*'s best alignment (7.12) and the *ClustalW* alignment for the sequences that belong to this cluster. The results are shown in Figure 7.20 and Figure 7.21. In Figure 7.22, I show the *RNAalifold* prediction of the *Rfam* seed alignment restricted to the sequences belonging to *RNAforester*'s best cluster.

I do the same experiments for the *TPP Riboswitch* and the *U1 spliceosomal RNA* and then discuss the results.

[3] In the extended forest representation the sequence alignment is the alignment of leaf nodes.

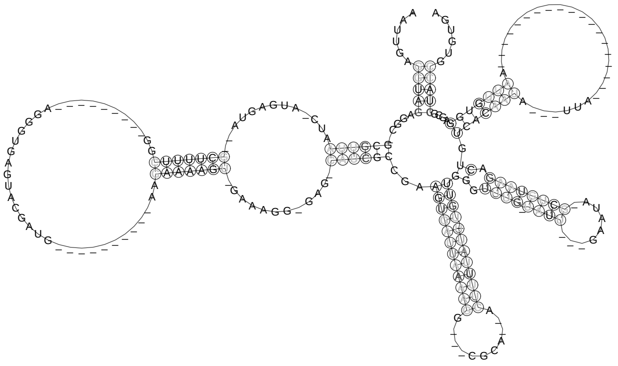


Figure 7.18: *Lysine Riboswitch. RNAalifold prediction for the seed alignment taken from the* Rfam *database. RNAalifold score:* $-37.70 = -22.44 + -15.26$*, Z:* $-3.1(-2.3)$.

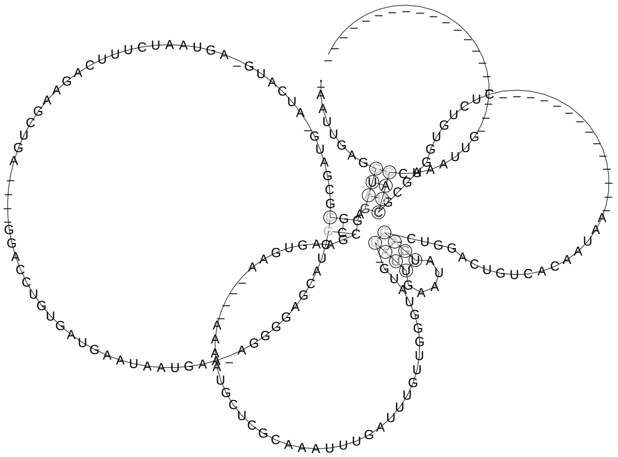


Figure 7.19: *Lysine Riboswitch. RNAalifold prediction for the ClustalW alignment of seed sequences taken from the Rfam database. RNAalifold score:* $-2.68 = -0.50 + -2.19$*, alifoldz score: n.a.*

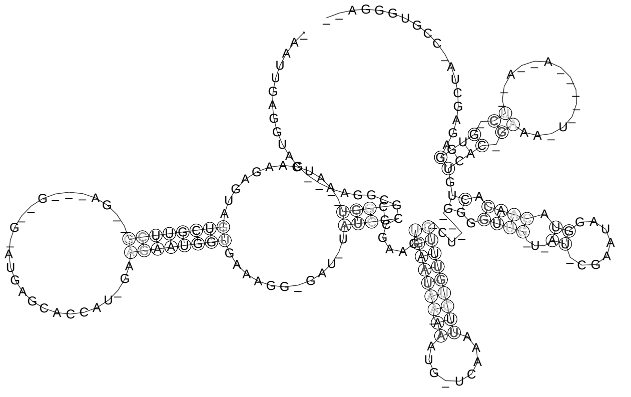


Figure 7.20: *Lysine Riboswitch. RNAalifold prediction for the RNAforester sequence alignment from the consensus structure in Figure 7.12. RNAalifold score:* $-25.65 = -12.04 + -13.62$*, alifoldz score:* $-1.9(-2.7)$*.*

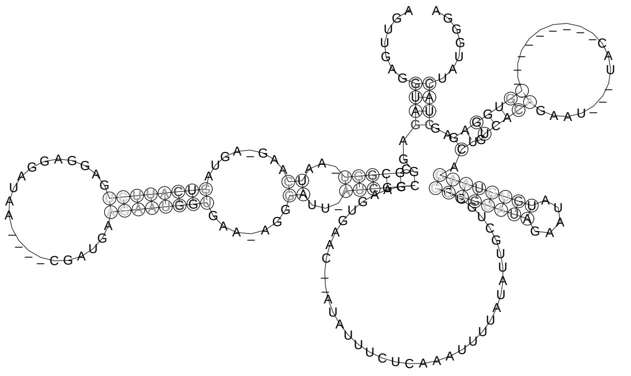


Figure 7.21: *Lysine Riboswitch. RNAalifold prediction for the ClustalW alignment for the sequences belonging to the consensus structure in Figure 7.12. RNAalifold score:* $-27.72 = -19.08 + -8.65$*, alifoldz score:* $-2.3(-1.1)$*.*

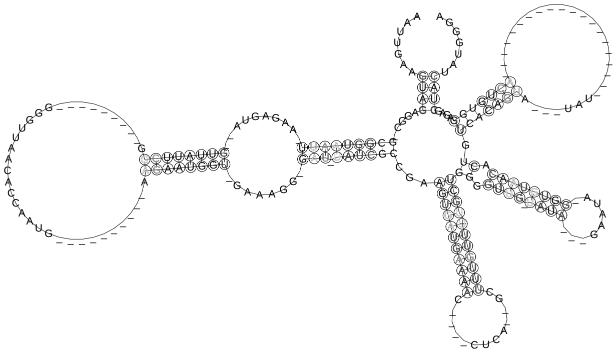


Figure 7.22: *Lysine Riboswitch. RNAalifold prediction for the Rfam seed alignment restricted to sequences belonging to the consensus structure in Figure 7.12. RNAalifold score:* $-44.05 = -25.75 + -18.29$, *alifoldz score:* $-6.8(-6.6)$.

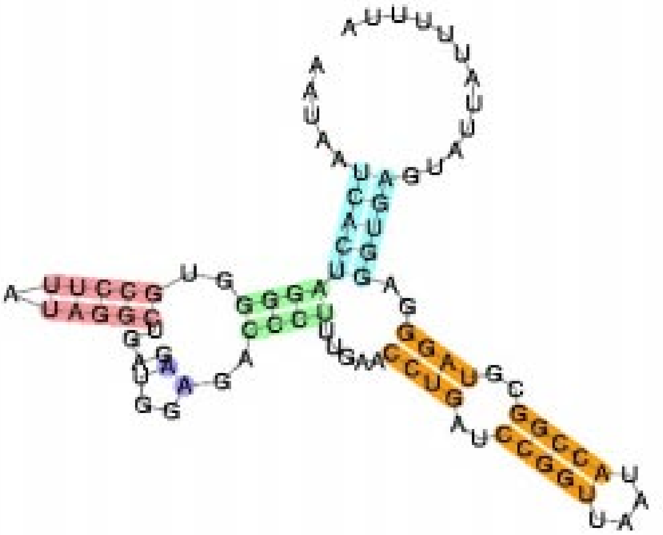


Figure 7.23: *Consensus structure of TPP riboswitch as published in [164].*

TPP Riboswitch (THI Element)

Vitamin B(1) in its active form thiamin pyrophosphate (TPP) is an essential coenzyme that is synthesized by coupling of pyrimidine and thiazole moieties in bacteria. The previously detected thiamin-regulatory element, thi box was extended, resulting in a new, highly conserved RNA secondary structure, the THI element, which is widely distributed in eubacteria and also occurs in some archaea [164].

The 141 sequences from the *Rfam* seed alignment for the *TPP riboswitch* (Accession number: RF00059) have an average length of 104.9 and an average identity of 52%. Figure 7.23 shows the consensus structure as published in *Rfam*. The Figures 7.24-7.29 show the structure predictions analog to the experiments for *Lysine Riboswitch*.

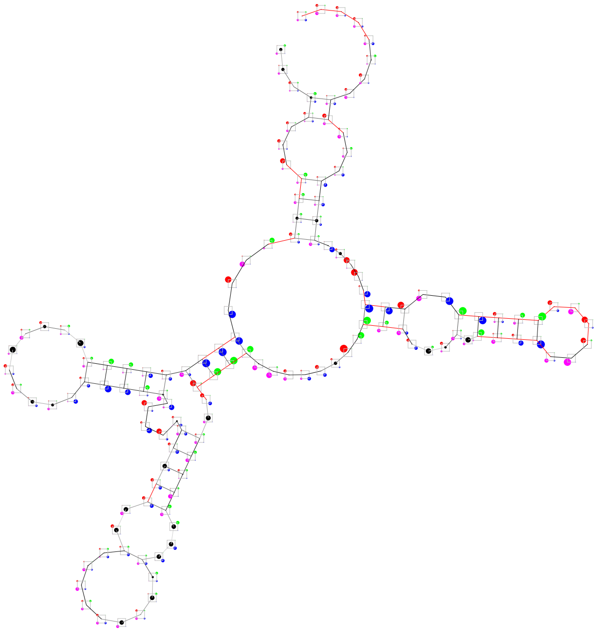


Figure 7.24: *TPP Riboswitch. Consensus structure of* 31 *sequences as predicted by RNAforester.*

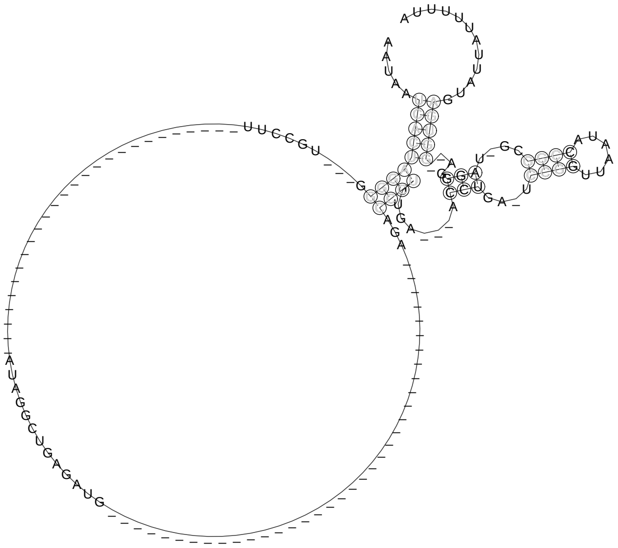


Figure 7.25: *TPP Riboswitch. RNAalifold prediction for the seed alignment taken from the Rfam database. RNAalifold score:* −12.11 = −9.26+−2.85, *alifoldz score:* 0(1.4).

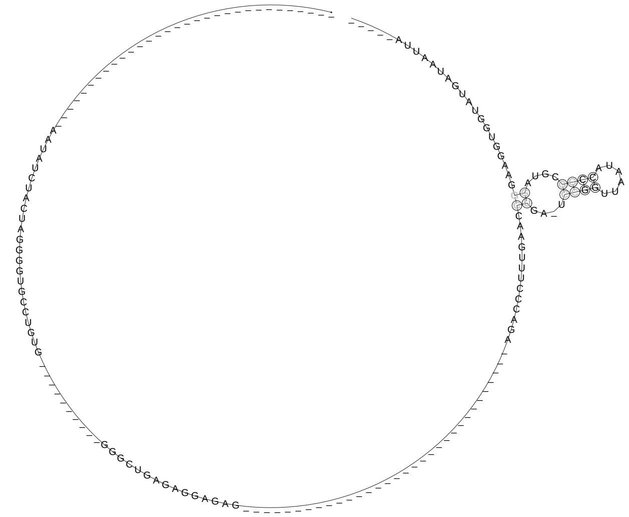


Figure 7.26: *TPP Riboswitch. RNAalifold prediction for the ClustalW alignment of seed sequences taken from the Rfam database. RNAalifold score:* $-5.64 = -4.33 + -1.31$*, alifoldz score:* $0.9(0.9)$.

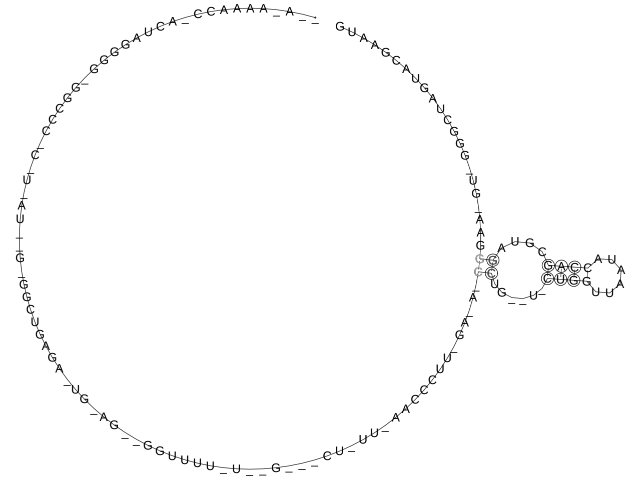


Figure 7.27: *TPP Riboswitch. RNAalifold prediction for the RNAforester sequence alignment for the consensus structure in Figure 7.24. RNAalifold score:* $-4.98 = -3.35 + -1.64$*, alifoldz score:* $-1.4(-1.1)$.

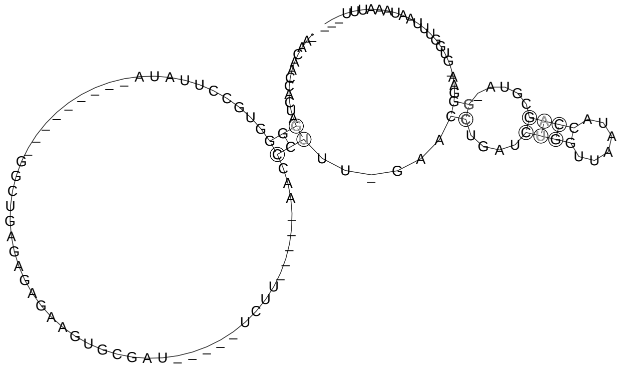


Figure 7.28: *TPP Riboswitch. RNAalifold prediction for the ClustalW alignment for the sequences belonging to the consensus structure in Figure 7.24. RNAalifold score:* $-6.81 = -5.29 + -1.53$*, alifoldz score:* $-1.7(-1.2)$.

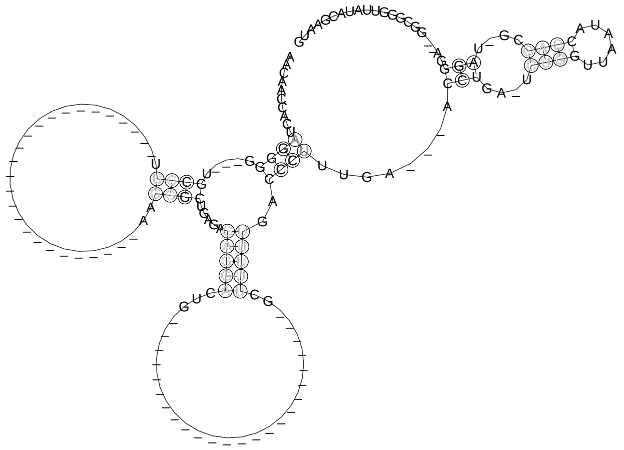


Figure 7.29: *TPP Riboswitch. RNAalifold prediction for the Rfam seed alignment restricted to sequences belonging to the consensus structure in Figure 7.24. RNAalifold score:* $-14.56 = -12.50 + -2.06$*, alifoldz score:* $-18.7(-12.3)$.

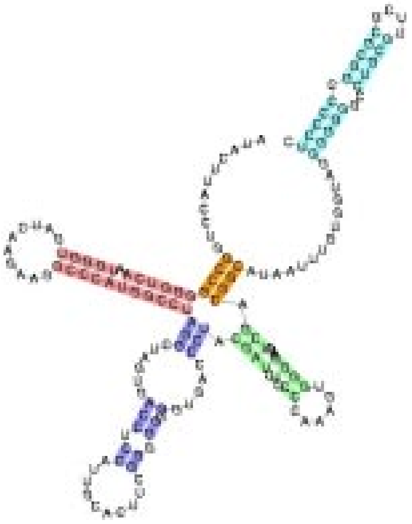


Figure 7.30: *Consensus structure of TPP riboswitch as published in [107].*

U1 spliceosomal RNA

U1 is a small nuclear RNA (snRNA) component of the spliceosome (involved in pre-mRNA splicing). Its 5' end forms complementary base pairs with the 5' splice junction, thus defining the 5' donor site of an intron. There are significant differences in sequence and secondary structure between metazoan and yeast U1 snRNAs, the latter being much longer (568 nucleotides as compared to 164 nucleotides in human). Nevertheless, secondary structure predictions suggest that all U1 snRNAs share a 'common core' [107].

The 54 sequences from the *Rfam* seed alignment for the *U1 spliceosomal RNA* (Accession number: RF00003) have an average length of 154.9 and an average identity of 59%. This family does not contain the larger yeast sequences. Figure 7.30 shows the consensus structure as published in *Rfam*. The Figures 7.31-7.36 show the structure predictions analog to the experiments for *Lysine Riboswitch*.

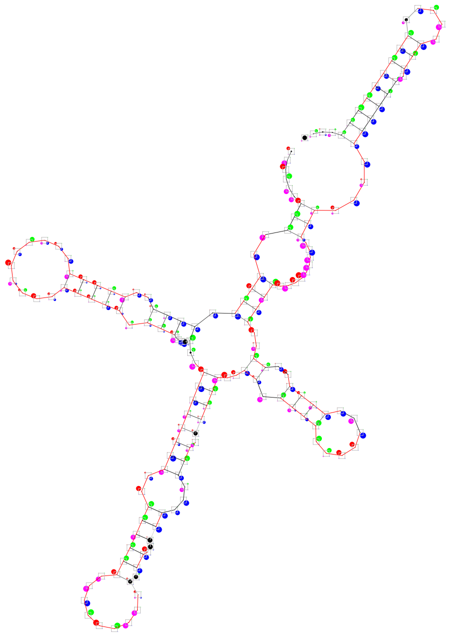


Figure 7.31: *U1 RNA. Consensus structure of* 14 *sequences as predicted by RNAforester.*

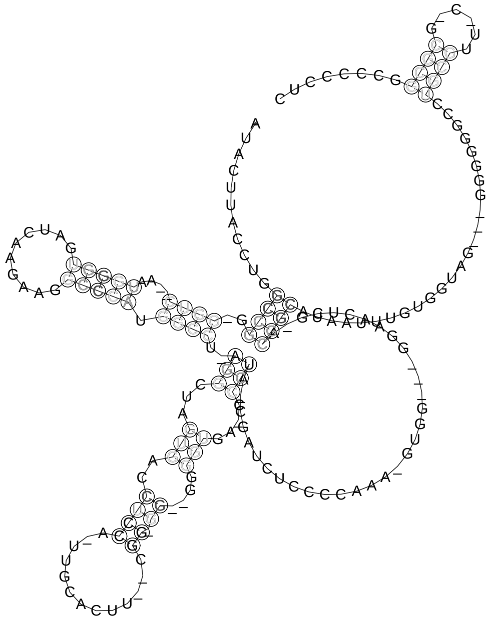


Figure 7.32: *U1 RNA. RNAalifold prediction for the seed alignment taken from the Rfam database. RNAalifold score:* $-23.27 = -14.69 + -8.58$*, alifoldz score:* 1.0(0.4).

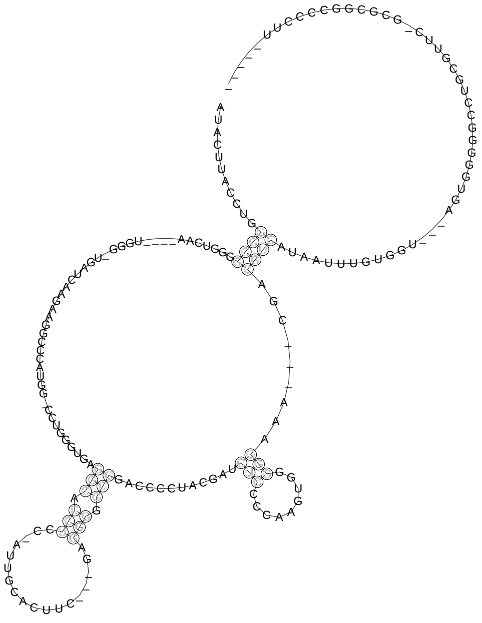


Figure 7.33: *U1 RNA. RNAalifold prediction for the ClustalW alignment of seed sequences taken from the Rfam database. RNAalifold score:* $-5.02 = -1.98 + -3.05$, *alifoldz score:* $0.9(n.v.)$.

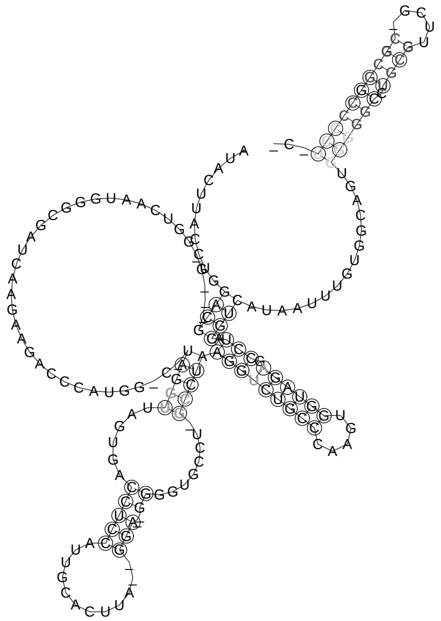


Figure 7.34: *U1 RNA. RNAalifold prediction for the RNAforester sequence alignment for the consensus structure in Figure 7.31. RNAalifold score:* $-36.50 = -31.85 + -4.65$, *alifoldz score Z:* $-5.3(-1.3)$.

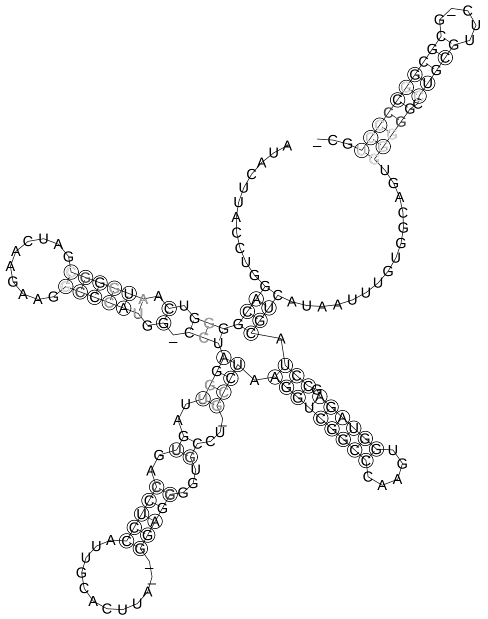


Figure 7.35: *U1 RNA. RNAalifold prediction for the ClustalW alignment for the sequences belonging to the consensus structure in Figure 7.31. RNAalifold score:* $-51.73 = -45.54 + -6.19$, *alifoldz score:* $-5.3(-2.6)$.

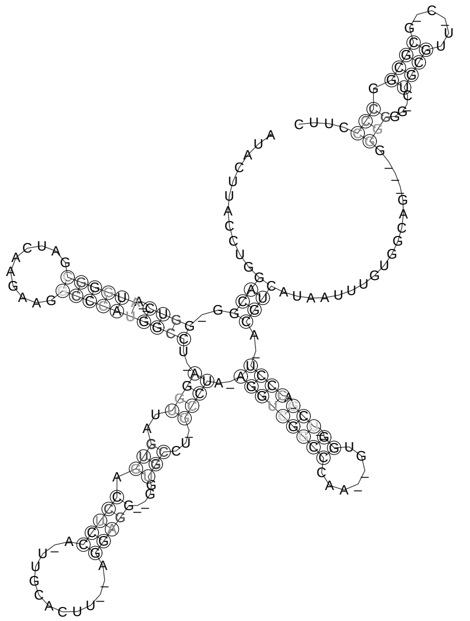


Figure 7.36: *U1 RNA. RNAalifold prediction for the Rfam seed alignment restricted to sequences belonging to the consensus structure in Figure 7.31. RNAalifold score:* $-34.18 = -27.36 + -6.82$, *alifoldz score:* $-7.8(-4.2)$.

Discussion

Evidently, the sequence alignment strategy is not a successful strategy to predict a consensus structure for RNA families that are distantly related (applied to the complete *Rfam* seed sequences). For the structure alignment strategy, the *RNAforester* cluster with the highest sequence diversity was always in good correspondence with the published consensus structure. The clusters that are not shown for the *TPP riboswitch* and the *U1 splicosomal RNA* were either diverse in their sequence and similar to the published structure[4], or similar in their sequence with a structural topology that is different to the published one. It seems to be unlikely that different sequences fold into a similar structure just by chance. Interestingly, *RNAalifold* was not able to repredict all stems of the consensus structure for the *TPP riboswitch* and the *U1 splicosomal RNA* for the hand-crafted seed alignments taken from the *Rfam* database.

To assess the quality of the sequence alignment that can be derived from *RNAforester*'s best cluster, I ran *RNAalifold* on the sequence alignment that was derived from the structural alignment. Additionally, I considered the *RNAalifold* predictions for the *ClustalW* alignment and the resticted seed alignment for the sequences belonging to this cluster. The predictions from the *ClustalW* alignments achieved a similar quality as the predictions from the *RNAforester* derived sequence alignments. However, the *RNAalifold* predictions detected different parts of the consensus structure. In particular, for the *Lysine riboswitch*, a stem that was detected with the *RNAforester* sequence alignment was not detected with the *ClustalW* alignment, and vice versa (see Figure 7.20 and 7.21). What remains is to observe whether the improved quality of the *ClustalW* alignments is simply due to a reduced sequence identity or a good pre-selection by *RNAforester*. In contrast to the unrestricted seed alignments, the restricted seed alignments let *RNAalifold* predict consensus structures that are in almost perfect correspondence to the published ones.

[4] A tuning of the *RNAforester* parameters could join them in a larger cluster.

My initial strategy was to use the zscores as a measure for the quality of the alignment. In contrast to my expectation, the zscores did not strongly identify the (unrestricted) seed alignments as an alignment of functional non-coding RNA sequences (A zscore below -4 would be a good indicator). The restricted seed alignments always achieved negative scores that gave strong evidence for a functional RNA.

Alignments of predicted minimal free energy structures can rightfully be criticized, because structure prediction may produce "optimal" structures quite different to the (suboptimal) native structure. The use of sequence similarity, if sufficient, is advocated as a means to avoid this dilemma. However, my experiments contribute two new considerations to this issue:

- They demonstrate an effect that, at the first sight, is paradoxical: strong sequence similarity can mislead the determination of the consensus structure. This happens because very similar sequences tend to fold into a similar structure, be it wrong or right.
- They demonstrate that a multiple structure alignment when applying the cutoff value in the clustering step, may produce meaningful alignments even in the presence of incorrect predictions.

As a consequence, a new approach to consensus construction becomes feasible, where first a good candidate consensus (or several) is constructed and subsequently, sequences that do not fall into a consensus cluster are refolded, given the candidate consensus as a target structure.

Chapter 8

Conclusions

In this thesis, I have analyzed the tree alignment model for the comparison of RNA secondary structures. I gave a systematic generalization of the alignment model from strings to trees and forests. I provided carefully engineered dynamic programming implementations using dense, two-dimensional tables which considerably reduces the space requirement. I introduced local similarity problems on forests and provided efficient algorithms that solve them. Since the problem of aligning trees occurs in many different disciplines, I untied my algorithmic contributions from the problem of aligning RNA secondary structures. For instance, using my algorithms I could contribute to address problems in the field of robotics [48, 165].

However, the main focus of this thesis is to provide algorithms to analyze RNA secondary structures. To improve the biological semantic of aligning RNA secondary structures as forests, I introduced an extended forest representation and a refined forest alignment model. The local similarity variants that were introduced on an abstract level of forests turned into local similarity notions for RNA secondary structures. The joined work with Thomas Töller showed that local structural motifs in RNA molecules can be successfully detected using my algorithms [203]. To make the results of structure comparison visually available, I invented a 2d-plot for RNA secondary structure alignments that highlights the differences and similarities of structures. This visualization is more intuitive than comparing abstract representations of RNA secondary structures, e.g. dot-plots, mountain plots, and makes it efficient to present results from structure comparison.

I generalized the forest alignment model to the case of multiple forests and, thus, made it applicable to compare multiple RNA secondary structures. My approach is a faithful generalization of established techniques used in sequence comparison. All the experience that has accumulated for multiple sequence alignments therefore carries over now to RNA secondary structures. I generalized the idea of sequence profiles to forests profiles, resulting in a profile of RNA secondary structures which groups different RNA secondary structures into a single data structure. To visualize a common consensus structure, I proposed a 2d-plot visualization that, in addition to structural similarity, can display the sequence diversity of the aligned structures. Based on these techniques, I proposed a consensus structure prediction strategy for families of RNA molecules that have low sequence homology. I demonstrated that this is a promising approach by successfully predicting the consensus structures for low sequence conserved RNA families taken from the *Rfam* database.

I implemented all algorithms presented in this thesis in the RNA structure comparison tool *RNAforester*. *RNAforester* is designed in spirit of the programs in the *Vienna RNA package* and will be distributed in the forthcoming *Vienna RNA Package Version 1.6*. The online version and the stand-alone application is publicly available at `http://bibiserv.techfak.uni-bielefeld.de/rnaforester`.

Future Work Several research activities open directly from the contributions in this thesis:

- The success of the structure prediction strategy that was presented in this thesis depends largely on the quality of thermodynamic predictions. It is well known that the biologically meaningful structure often hides in the space of suboptimal solutions. I argue that results of my structure prediction strategy can be improved significantly by considering suboptimal solutions. However, the exponential number of suboptimal solutions prohibits a straightforward strategy. Recently, Giegerich et al. provided the structure prediction program *RNAshapes* based on thermodynamics that compartmentalizes the suboptimal solution space into different *shapes* [59]. A combination of *RNAshapes* and *RNAforester* is the logically next step.

- That locally similar structures can be detected with *RNAforester* without prior knowledge was demonstrated in this thesis. The application of my algorithms on a genome-wide scale is a challenging task. Locally stable structures could be predicted on genome-wide surveys using *RNALfold* [85] and the resulting data could be analyzed for locally conserved structures using *RNAforester*, after it has been preprocessed for length and energy constraints. Thorough statistics have to be done to rank the locally conserved structures and distinguish biologically relevant conservations from those that are found just by chance.
- A well known problem of the progressive strategy is that errors made early in an alignment cannot be rectified when further sequences are added. Notredame et al. present a strategy that can minimize this effect in the multiple sequence alignment tool *T-Coffee* [149]. Instead of using substitution scores for the calculation of pairwise alignments, they propose a position dependent scoring. A primary library gathers information from heterogeneous sources for pairwise alignments, such as sequence alignments (global and local), structural alignments and manual alignments. These sources are combined in an extended library such that each pair of characters in the sequences has a position specific weight. The pairwise alignments are then optimized according to this extended library. Misplacing gaps in the earlier steps of the progressive calculation become less likely and significantly improves the quality of the alignment in comparison to *ClustalW* and other tools. An analogous strategy for trees could further improve the quality of multiple tree alignments.

- Various tree distances have been discussed in the introductory chapter of this thesis. However, a thorough analysis of their quality for RNA secondary structures is missing. It would be interesting to observe whether, and under which circumstances, the distances can be replaced by each other and provide similar results. The complexities of the tree distances depend on different parameters of the tree structure, e.g. the number of nodes, the depth, the number of leaves, and the degree. All these parameters are known and, thus, the computational effort can be determined in advance. At the end, a flexible strategy could always chose the "cheapest" model.

- Today, the detection of unknown non-coding RNA from genomic data is one of the biggest challenges in molecular biology. First successes were achieved with tools that infer a structure from a (multiple) sequence alignment by thermodynamic and phylogenetic information, comparing the result of the predictions with randomized data [163, 223]. However, there is an inherent problem: If the sequences are highly conserved, the alignment is good but the covariance of base-paired regions is low. Thus, the thermodynamic considerations dominate the structure prediction. Unlike stated by Maizel and coworkers, energy seems not to be a good discriminator to separate structural from non structural RNAs [19, 162]. If the sequence conservation is too low, regions of covariance are not aligned accurate and the alignment can mislead the predictions. As in my structure prediction strategy, I am thinking about a strategy that goes the other way around: I could start with thermodynamic considerations and then use phylogenetic information to estimate the reliability of predictions.

A multiple and local structure alignment program will become a basic tool, just like the sequence counterparts. With *RNAforester*, I provide a program that can be embedded in a larger framework of structure analysis, contributing to solve problems beyond the ones I proposed.

Bibliography

[1] K Akoi. A top-down algorithm to compute the distance between trees. *Trans. IECE*, 66:49–56, 1983. (in Japanese).

[2] J. Alber, J. Gramm, J. Guo, and R. Niedermeier. Computing the similarity of two sequences with nested arc annotations. *Theoretical Computer Science*, 312:337 – 358, 2004.

[3] L. Alonso and R. Schott. On the tree inclusion problem. *Acta Informatica*, 37:653–670, 2001.

[4] V. Bafna, S. Muthukrishnan, and R. Ravi. Computing similarity between RNA strings. In *Proc. of* CPM'95*,* LNCS 937, pages 1–16, 1995.

[5] D. Baltimore. RNA-dependent DNA polymerase in virions of RNA tumour viruses. *Nature*, 226:1209–1211, 1970.

[6] D. Barash. Second eigenvalue of the Laplacian matrix for predicting RNA conformational switch by mutation. *Bioinformatics*, 20(12):1861–1869, 2004.

[7] D.T. Barnard, G. Clarke, and N. Duncan. Tree-to-tree Correction for Document Trees. Technical report, Queen's University, Department of Computing and Information Science, Kingston, Ontario K7L 3N6, Canada, January 1995.

[8] R.T Batey and J.A. Doudna. The parallel universe of RNA folding. *Nature struct. Biol.*, 5:337–340, 1998.

[9] R. Bellman. *Dynamic Programming.* Princeton University Press, 1957.

[10] P. Billi. Ordered Tree Edit Distance with Merge and Split Operations. Technical report, IT University of Copenhagen, September 2003. ISSN 1600-6100, ISBN 87-7949-048-4.

[11] P. Billi. Tree Edit Distance, Alignment Distance and Inclusion. Technical report, IT University of Copenhagen, 2003. ISSN 1600-6100, ISBN 87-7949-032-8.

[12] P. Bonizzoni and G. Della Vedova. The Complexity of Multiple Sequence Alignment with SP-Score that is a Metric. *Theoretical Computer Science*, 259(1):63–79, 2001.

[13] P. Brion and E. Westhof. Hirarchy and dynamics of RNA folding. *A. Rev. Biophys. Bioimol. Struct.*, 26:113–137, 1997.

[14] R.E. Bruccoleri and G. Heinrich. An improved algorithm for nucleic acid secondary structure display. *Comp. Appl. Biosci.*, 4:167–173, 1988.

[15] H. Carillo and D.J. Lipman. The multiple sequence alignment problem in biology. *SIAM Journal of Applied Mathematics*, 48:1073–1082, 1988.

[16] J.C. Carrington and V. Ambros. Role of microRNAs plant and animal development. *Science*, 301:336–338, 2003.

[17] T.R. Cech. RNA as an enzyme. *Scientific American*, 5:64–75, 1986.

[18] G.J.S. Chang, G. Patel, L. Relihan, and J.T.L. Wang. A graphical environment for change detection in structured documents. In R. Baeza-Yates et al., editor, *Proceedings of the 21st Annual International Computer Software and Applications Conference (COMPSAC'97)*, pages 536–541, Washington, DC, August 1997.

[19] J.H. Chen, B. Shapiro, K.M. Currey, and J.V Maizel. A computational procedure for assessing the significance of RNA secondary structure. *Bioinformatics*, 6:7–18, 1990.

[20] W. Chen. More efficient algorithm for ordered tree inclusion. *Journal of Algorithms*, 26:370–385, 1998.

[21] W. Chen. New algorithm for ordered tree-to-tree correction problem. *Journal of Algorithms*, 40:135–158, 2001.

[22] Y.C. Cheng and S.Y. Lu. Waveform correlation by tree matching. In *Proceedings of the 9th Annual International Computer Software and Application Conference*, pages 299–305, 1985.

[23] R. Chenna, H. Sugawara, T. Koike, R. Lopez, T.J Gibson, D.G. Higgins, and J.D. Thompson. Multiple sequence alignment with the clustal series of programs. *Proc. Nat. Acad. Sci. U.S.A.*, 31(13):3497–3500, 2003.

[24] F. Chetouani, P. Monestie, P. Thebault, C. Gaspin, and B. Michot. ESSA: an integrated and interactive computer tool for analysing RNA secondary structure. *Nucleic Acids Res.*, 25(17):3514–3522, 1997.

[25] C. Chevalet and B. Michot. An algorithm for comparing RNA secondary structures and search for similar substructures. *Comp. Appl. Biosci.*, 8(3):215–225, 1992.

[26] D.K.Y. Chiu and T. Kolodziejczak. Inferring consensus structure from nucleic acid sequences. *Comp. Appl. Biosci.*, 7:347–352, 1991.

[27] S. Chowdhury, C. Ragaz, E. Kreuger, and F. Narberhaus. Temperature-controlled Structural Alterations of an RNA Thermometer. *J. Biol. Chem.*, 278:47915–47921, 2003.

[28] R. Cole, R. Hariharan, and P. Indyk. Tree pattern matching and subset matching in deterministic $O(n \log^3 n)$ time. In *ACM-SIAM Symposium on Discrete Algorithms (SODA'99)*, pages 245–254, 1999.

[29] G. Collins, S.Y. Le, and K. Zhang. A new algorithm for computing similarity between RNA structures. *Information Sciences*, 139:59–77, 2001.

[30] F. Crick. Central Dogma of Molecular Biology. *Nature*, 227:561–563, 1970.

[31] M.O. Dayhoff, R.M. Schwartz, and B.C Orcutt. A model of evolutionary change in proteins. In M.O. Dayhoff, editor, *Atlas of Protein Sequence and Structure*, volume 5, pages 345–352. National Biomedical Research Foundation, 1978.

[32] P. de Rijk, Y.V. de Peer, S. Chapelle, and R. de Wachter. Database on structure of small ribosomal subunit RNA. *Nucleic Acids Res.*, 22:3495–3501, 1994.

[33] P. de Rijk and R. de Wachter. RnaViz, a program for the visualisation of RNA secondary structure. *Nucleic Acids Res.*, 25(22):4679–4684, 1997.

[34] P. de Rijk, J. Wuyts, and R. de Wachter. RnaViz2: an improved representation of RNA secondary structure. *Bioinformatics*, 19(2):299–300, 2003.

[35] R. Dirks and N. Pierce. A partition function algorithm for nucleic adic secondary struture, including pseudoknots. *Journal of Computational Chemistry*, 24:1664–1677, 2003.

[36] P. Doty, H. Boedtker, J.R. Fresco, R. Haselkorn, and M. Litt. Secondary structure in ribonucleic acids. In *Proc. Nat. Acad. Sci. U.S.A.*, volume 45, pages 482–499, 1959.

[37] D. Dowell and S.R. Eddy. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics*, 5(71), 2004.

[38] M. Dubiner, Z. Galil, and E. Magen. Faster Tree Pattern Matching. In *Proc. of* FOCS'90, pages 145–150, 1990.

[39] S. Dulucq and L. Tichit. RNA secondary structure comparison: exact analysis of the Zhang-Sasha tree edit-algorithm. *Theoretical Computer Science*, 306:471–484, 2003.

[40] S. Dulucq and H. Touzet. Analysis of Tree Edit Distance Algorithms. In *Proc. of* CPM'03*,* LNCS 2676, pages 83–95. Springer-Verlag, 2003.

[41] R. Durbin, S. Eddy, A. Krogh, and G. Mitchinson, editors. *Biological Sequence Analysis*. Cambridge University Press, UK, 1998.

[42] S. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Res.*, 22(11):2079–2088, 1994.

[43] S.R. Eddy. Non-Coding RNA Genes and the Modern RNA World. *Nature Reviews Genetics*, 2(12):919–929, 2001.

[44] A. Edvardsson, P. Gardner, A.M. Poole, M.D. Hendy, D. Penny, and V. Moulton. A Search for H/ACA SnoRNAs in Yeast Using MFE Secondary Structure Prediction. *Bioinformatics*, 19(7):865–873, 2003.

[45] P. Evans. *Algorithms and Complexity for Arc Annotated Sequence Analysis*. PhD thesis, University of Victoria, 1999.

[46] D. Feng and F. Doolittle. Progressive sequence alignment as a prerequisite to correct to correct phylogenetic trees. *Journal of Molecular Evolution*, 25:351–360, 1987.

[47] D. Fera, N. Kim, N. Shiffeldrim, J. Zorn, U. Laseron, H.H. Gan, and T. Schlick. RAG: RNA-As-Graphs web resource. *BMC Bioinformatics*, 5:88, 2004.

[48] M. Ferch, J. Zhang, and M. Höchsmann. Learning cooperative assembly with the graph representation of a state-action space. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[49] C. Flamm, W. Fontana, I.L. Hofacker, and P. Schuster. RNA folding at elementary step resolution. *RNA*, 6:325–338, 2000.

[50] J.R. Fresco, B.M. Alberts, and P. Doty. Some molecular details of the secondary structure of ribonucleic acid. *Nature*, 188:98–101, 1960.

[51] H. Gabow. *Implementation of Algorithms for Maximum Matching on Nonbipartite Graphs*. PhD thesis, Stanford University, 1973.

[52] H.H. Gan, D. Fera, J. Zorn, N. Shiffeldrim, M. Tang, U. Laserson, N. Kim, and T. Schlick. RAG: RNA-As-Graphs database–concepts, analysis, and features. *Bioinformatics*, 20(8):1285–1291, 2004.

[53] H.H. Gan, S. Pasquali, and T. Schlick. Exploring the repertoire of RNA secondary motifs using graph theory with implications for RNA design. *Nucleic Acids Res.*, 31:2926–2943, 2004.

[54] P. Gardner. *Simulating the RNA-world and Computational Ribonomics*. PhD thesis, Massey University, Palmerston North, New Zealand., 2003.

[55] P.P. Gardner and R. Giegerich. A comprehensive comparison of comparative RNA structure prediction approaches. *BMC*, 5(1):140, 2004.

[56] R. Giegerich. A Systematic Approach to Dynamic Programming in Bioinformatics. *Bioinformatics*, 16:665–677, 2000.

[57] R. Giegerich and D. Evers. RNA Movies: visualizing RNA secondary structure spaces. *Bioinformatics*, 15(1):32–37, 1999.

[58] R. Giegerich and C. Meyer. Algebraic dynamic programming. In Hélène Kirchner and Christophe Ringeissen, editors, *Algebraic Methodology And Software Technology, 9th International Conference, AMAST 2002*, pages 349–364, Saint-Gilles-les-Bains, Reunion Island, France, 2002. Springer LNCS 2422.

[59] R. Giegerich, B. Voss, and M. Rehmsmeier. Abstract Shapes of RNA. *Nucleic Acids Res.*, 20:4843–4851, 2004.

[60] W. Gilbert. Origin of Life, The RNA World. *Nature*, 319:618, 1986.

[61] GNU Build Tools: autoconf and automake. www.gnu.org.

[62] O. Gotoh. Significant improvement in accuracy of multiple protein sequence alignments by iterative refinements as assessed by reference to structural alignments. *J. Mol. Biol.*, 264:823–838, 1996.

[63] O. Gotoh. Multiple sequence alignment: algorithms and applications. *Adv Biophys.*, 36:159–206, 1999.

[64] Graphviz - Graph Visualization Software. www.graphviz.org.

[65] N.K. Gray and M. Wickens. Control of Translation Initiation in Animals. *Annu. Rev. Cell Dev. Biol.*, 14:399–458, 1998.

[66] M. Gribskov, A. McLachlan, and D. Eisenberg. Profile analysis detection of distantly related Proteins. In *Proc. Nat. Acad. Sci. U.S.A.*, volume 88, pages 55–58, 1987.

[67] S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S.R. Eddy. Rfam: an RNA family database. *Nucleic Acids Res.*, 31(1):439–441, 2003.

[68] S. Griffiths-Jones, S. Moxon, M. Marshall, A. Khanna, Eddy S.R., and A. Bateman. Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res.*, 33:D121–D124, 2005.

[69] J. Guhaniyogi and G. Brewer. Regulation of mRNA stability in mammalian cells. *Gene*, 265:11–23, 2001.

[70] A.P. Gultyaev, F.H.D. vanbatenburg, and C.W.A. Pleij. The computer simulation of RNA folding pathways using a genetic algorithm. *J. Mol. Biol.*, 250:37–51, 1995.

[71] D. Gusfield, K. Balasubramanian, and D. Naor. Parametric optimization of sequence alignment. *Algorithmica*, 12:312–326, 1994.

[72] K. Han and Y. Byun. PSEUDOVIEWER2: visualization of RNA pseudoknots of any type. *Nucleic Acids Res.*, 31(13):3432–3440, 2003.

[73] K. Han and H.-J. Kim. Prediction of common folding structures of homologous RNAs. *Nucleic Acids Res.*, 21:1251–1257, 1993.

[74] K. Han, Y. Lee, and W. Kim. PseudoViewer: automatic visualization of RNA pseudoknots. *Bioinformatics*, 18(1):321–328, 2002.

[75] S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from protein blocks. In *Proc. Nat. Acad. Sci. U.S.A.*, volume 89, pages 10915–9, 1992.

[76] D.S. Hirschberg. Algorithms for the Longest Common Subsequence Problem. *Journal of the ACM*, 24(2):664 – 675, 1977.

[77] M. Höchsmann, T. Töller, R. Giegerich, and S. Kurtz. Local Similarity in RNA Secondary Structures. In *Proc. of* CSB'03, pages 159–168, 2003.

[78] M. Höchsmann, B. Voss, and R. Giegerich. Pure Multiple RNA Secondary Structure Alignments: A Progressive Profile Approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):53–62, 2004.

[79] I.L. Hofacker. Vienna RNA secondary structure server. *Nucleic Acids Res.*, 31(13):3429–3431, 2003.

[80] I.L. Hofacker, H.F. Bernhart, and P.F. Stadler. Alignment of RNA Base Pairing Probability Matrices. *Bioinformatics*, 20:2222–2227, 2004.

[81] I.L. Hofacker, M. Fekete, C. Flamm, M.A. Huynen, A. Rauscher, P.E. Stolorz, and P.F. Stadler. Automatic Detection of Conserved RNA Structure Elements in Complete RNA Virus Genomes. *Nucleic Acids Res.*, 26:3825–3836, 1998.

[82] I.L. Hofacker, M. Fekete, and P. Stadler. Secondary structure prediction for aligned RNA sequences. *J. Mol. Biol.*, 319(5):1059–1066, 2002.

[83] I.L. Hofacker, M. Fekete, and P.F. Stadler. Secondary Structure Prediction for Aligned RNA Sequences. *J. Mol. Biol.*, 319:1059–1066, 2002.

[84] I.L. Hofacker, W. Fontana, P.F. Stadler, S. Bonhoeffer, M. Tacker, and P. Schuster. Fast Folding and Comparison of RNA Secondary Structures. *Monatshefte f. Chemie*, 125:167–188, 1994.

[85] I.L. Hofacker, B. Priwitzer, and P.F. Stadler. Prediction of Locally Stable RNA Secondary Structures for Genome-wide Surveys. *Bioinformatics*, 20:191–198, 2004.

[86] C.M. Hoffmann and M.J. O'Donnell. Pattern Matching in Trees. *Journal of the ACM*, 29:68–95, 1982.

[87] P. Hogeweg and B. Hesper. Energy directed folding of RNA sequences. *Nucleic Acids Res.*, 12:67–74, 1984.

[88] P. Hogeweg and B. Hesper. The alignment of sets of sequences and the construction of phyletic trees: an integrated method. *J. Mol. Biol.*, 20(2):175–186, 1984.

[89] S.R. Holbrook and S.H. Kim. RNA crystallography. *Biopolymers*, 44:3–21, 1997.

[90] R.C. Holt and J.R. Cordy. The Turing Programming Languag. *Communications of the ACM*, 31(12):1410–1423, 1988.

[91] R.P. Jansen. mRNA Localization: Message On The Move. *Nature Rev. Mol. Cell Biol.*, 2:247–256, 2001.

[92] J. Jansson and A. Lingas. A fast algorithm for optimal alignment between similar ordered trees. *Fundamenta Informaticae, Special issue on computing patterns in strings*, 56(1,2):105–120, 2003.

[93] T. Jiang, G. Lin, B. Ma, and K. Zhang. The longest common subsequence problem for arc-annotated sequences. *J. Discrete Algorithms*, 2(2):257–270, 2004.

[94] T. Jiang, G.-H. Lin, B. Ma, and K. Zhang. A General Edit Distance between RNA Structures. *J. Comp. Biol.*, 9(2):371–388, 2002.

[95] T. Jiang, J.T.L. Wang, and K. Zhang. Alignment of Trees - An Alternative to Tree Edit. *Theoretical Computer Science*, 143(1):137–148, 1995.

[96] P. Kilpeläinen. *Tree Matching Problems with Applications to Structured Text Databases.* PhD thesis, University of Helsinki, Department of Computer Science, November 1992.

[97] P. Kilpeläinen and H. Mannila. Grammatical Tree Matching. In *Proc. of* CPM'92*,* LNCS 644, pages 159–171, 1992.

[98] P. Kilpeläinen and H. Mannila. Ordered and Unordered Tree Inclusion. *SIAM Journal on Computing*, 24(2):340–356, 1995.

[99] S.H. Kim, G.J. Suddath, G.J. Quigley, A. McPherson, J.L. Sussmann, A.H.J. Wang, N.C. Seeman, and A. Rich. Three-dimensional tertiary structure of yeast phenylalanine transfer RNA. *Science*, 185:435–439, 1974.

[100] J. Kjems and J. Egebjerg. Modern methods for probing RNA structure. *Cur. Opin. Biotech*, 9:59–65, 1996.

[101] R.D. Klausner, T.A. Rouault, and J.B. Harford. Regulating the fate of mRNA: The Control of Cellular Iron Metabolism. *Cell*, 72:19–28, 1993.

[102] P. Klein. Computing the Edit-Distance between Unrooted Ordered Trees. In *Proc. of* ESA'98, LNCS 1461, pages 91–102. Springer-Verlag, 1998.

[103] R.J. Klein and S.R. Eddy. RSEARCH: Finding homologs of single structured RNA sequences. *BMC Bioinformatics*, 4(44), 2003.

[104] B. Knudsen and J. Hein. RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res.*, 31(13):3423–8, 2003.

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

[106] S.R. Kosaraju. Efficient Tree Pattern Matching. In *Proc. of* FOCS'89, pages 178–183, 1989.

[107] L. Kretzner, A. Krol, and M. Rosbash. Saccharomyces cerevisiae U1 small nuclear RNA secondary structurecontains both universal and yeast-specific domains. In *Proc. Nat. Acad. Sci. U.S.A.*, volume 87, pages 851–855, 1990.

[108] S. Kurtz. Foundations of Sequence Analysis, 2000. Lecture notes for a course in the Wintersemester 2000/2001, Technische Fakultät, Universität Bielefeld.

[109] G. Lapalme, R.J. Cedergren, and D. Sankoff. An algorithm for the display of nucleic acid secondary structure. *Nucleic Acids Res.*, 10:8351–8356, 1982.

[110] S.Y. Le, R. Nussinov, and J.V Mazel. Tree graphs of RNA secondary structures and their comparison. *Computational Biomedical Research*, 22:461–473, 1989.

[111] S.Y. Le, J. Owens, R. Nussinov, J.H Chen, B. Shapiro, and J.V Mazel. RNA secondary structures: Comparisons and determinations of frequently recuring substructures by consensus. *Comp. Appl. Biosci.*, 5:205–210, 1989.

[112] S.Y. Le and M. Zuker. Predicting common foldings of homologous RNA. *J Biomol Struct Dyn.*, 8(5):1027–1044, 1991.

[113] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Dokl. Akad. Nauk USSR*, 163:845–848, 1965. (Russian), English Translation (1966), Cybernetics and Control Theory, 10, 707-710.

[114] G.-H. Lin, Z.-Z. Chen, T. Jiang, and J. Wen. The longest common subsequence problem for sequences with nested arc annotations. In *Proc. of* ICALP'01, pages 444–455, 2001.

[115] D.J. Lipman, S.F. Altschul, and J.D. Kececioglu. A tool for multiple sequence alignment. *Proc. Nat. Acad. Sci. U.S.A.*, 86:4412–4415, 1989.

[116] T.M. Lowe. GtRDB: the genomic tRNA database. World Wide Web, 2002. GtRDB: the genomic tRNA database.

[117] T.M. Lowe and S.R. Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.*, 25(5):955–964, 1997.

[118] S.Y. Lu. A tree-to-tree distance and its application to cluster analysis. *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 1:219–224, 1979.

[119] F. Luccio, A. Mesa, P. Olivares, and L. Pagli. Exact rooted subtree matching in sublinear time. In *Proc. of the 9-th. ANaCC/ACM/IEEE International Congress on Computer Science Research (CIICC'02)*, pages 27–35, 2002.

[120] R. Lück, S. Gräf, and G. Steger. ConStruct: A tool for thermodynamic controlled prediction of conserved secondary structure. *Nucleic Acids Res.*, 21:42084217, 1999.

[121] R. Lück, G. Steger, and D. Riesner. Thermodynamic prediction of conserved secondary structure: Application to RRE-element of HIV, tRNA-like element of BMV, and mRNA of prion protein. *J. Mol. Biol.*, 258:813–826, 1996.

[122] R.B. Lyngs and C.N.S. Pedersen. Pseudoknot prediction in energy based models. *J. Comp. Biol.*, 7:409 – 427, 2000.

[123] B. Ma, L. Wang, and K. Zhang. Computing similarity between RNA structures. *Theoretical Computer Science*, 276:111–132, 2002.

[124] D. Maier. The complexity of some problems on subsequence and supersequence. *Journal of the ACM*, 25(2):322 – 336, 1978.

[125] E. Mäkinen. On the subtree isomorphism problem for ordered trees. *Information Processing Letters*, 32:271–273, 1989.

[126] M. Mandal, B. Boese, J.E. Barrick, W.C. Winkler, and R.R. Breaker. Riboswitches control fundamental biochemical pathways in Bacillus subtilis and other bacteria. *Cell*, 113(5):577–586, 2003.

[127] A. Marian. Detecting Changes in XML Documents. In *Proceedings of the 18th International Conference on Data Engineering (ICDE'02)*, page 41, 2002.

[128] O. Martinez and B. Goud. Rab proteins. *Biochim. Biophys. Acta*, 1404:101–112, 1998.

[129] D. Mathews, J. Sabina, M. Zuker, and H. Turner. Expended sequence dependence of thermodynamic parameters provides robust prediction of RNA secondary structures. *J. Mol. Biol.*, 288:911–940, 1999.

[130] D.H. Mathews. Predicting a set of minimal free energy RNA secondary structures common to two sequences. *Bioinformatics*, Advance Online Access, 2005.

[131] D.H. Mathews and D.H. Turner. Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. *J. Mol. Biol.*, 317(2):191–203, 2002.

[132] O. Matzura and A. Wennborg. RNAdraw: an integrated program for RNA secondary structure calculation and analysis under 32-bit Microsoft Windows. *Comp. Appl. Biosci.*, 12(3):247–249, 1996.

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

[134] J.P. McCutcheon and S.R. Eddy. Computational identification of noncoding RNAs in Saccharomyces cerevisiae by comparative genomics. *Nucleic Acids Res.*, 31(4):4119–4128, 2003.

[135] C. Meyer and R. Giegerich. Matching and Significance Evaluation of combined sequence-structure motifs in RNA. *Z. Phys. Chem.*, 216:193–216, 2002.

[136] L. Milanovic and H. Wagner. g2 - graphic library. www.gnu.org.

[137] L. Milanovic and H. Wagner. MATHPROG: Solver for the Maximum Weight Matching Problem. http://elib.zib.de/pub/Packages/mathprog/matching/weighted/.

[138] A.A. Mironov, L.P. Dyakonova, and A.E. Kister. A kinetic approach to the prediction of RNA secondary structures. *J Biomol Struct Dyn.*, 2:953, 1985.

[139] B. Moayer and K.S. Fu. eA tree system approach for fingerprint pattern recognition. *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 8:376–387, 1986.

[140] B. Morgenstern. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15:211–218, 1999.

[141] B. Morgenstern, A. Dress, and T. Werner. Multiple DNA and protein sequence alignment based on segment-to-segment comparison. In *Proc. Nat. Acad. Sci. U.S.A.*, pages 12098–12103, 1996.

[142] V. Moulton, M. Zuker, M. Steel, R. Pointon, and D. Penny. Metrics on RNA structures. *J. Comp. Biol.*, 7(1):277–292, 2000.

[143] A. Nakata, K. Taura, K. Yamamoto, and A. Yonezawa. Visualization of RNA secondary structures using highly parallel computers. *Comp. Appl. Biosci.*, 12(3):205–211, 1996.

[144] A. Nierman and H.V. Jagadish. Evaluating structural similarity in XML documents. In *Proceedings of the 5th International Workshop on the Web and Databases (WebDB'02)*, 2002.

[145] A. Nocker, T. Hausherr, S. Balsiger, N.P. Krstulovic, H. Hennecke, and Narberhaus F. A mRNA-based thermosensor controls expression of rhizobial heat shock genes. *Nucleic Acids Res.*, 29:4800–4807, 2001.

[146] A. Nocker, N. P. Krstulovic, X. Perret, and F. Narberhaus. ROSE elements occur in disparate rhizobia and are functionally interchangeable between species. *Arch. Microbiol.*, 176:44–51, 2001.

[147] H.F. Noller and C.R. Woese. Secondary structure of 16S ribosomal RNA. *Science*, 212:403–41, 1981.

[148] C. Notredame. Recent progress in multiple sequence alignment: a survey. *Pharmacogenomics*, 3(2):131–144, 2002.

[149] C. Notredame, D.G. Higgins, and J. Heringa. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.*, 302:205–217, 2000.

[150] C. Notredame and D.G. Hiigins. SAGA: Sequence Alignment by Genetic Algorithm. *Nucleic Acids Res.*, 24:1515–1524, 1996.

[151] R. Nussinov, G. Pieczenik, J.R. Griggs, and D.J. Kleitman. Algorithms for loop matchings. *SIAM Journal of Applied Mathematics*, 35:68–82, 1979.

[152] B. Onoa and C. Bustamente. RNA folding and unfolding. *Curr. Opin. Struct. Biol.*, 14(3):374–379, 2004.

[153] J. Perochon-Dorisse, F. Chetouani, S. Aurel, N. Iscolo, and B. Michot. RNA_d2: a computer program for editing and display of RNA secondary structures. *Comp. Appl. Biosci.*, 11:101–109, 1995.

[154] G. Pesole, S. Liuni, G. Grillo, F. Licciulli, F. Mignone, C. Gissi, and C. Saccone. UTRdb and UTRsite: specialized database of sequences and functional elements of 5' and 3' untranslated regions of eukaryotic mRNAs. *Nucleic Acids Res.*, 30:335–340, 2002.

[155] S. Peyton Jones, editor. *Haskell 98 Language and Libraries – The Revised Report.* Cambridge University Press, April 2003.

[156] A.M. Pyle and J.B. Green. RNA folding. *Cur. Opin. Biotech*, 5:303–310, 1995.

[157] R. Ramesh and R. Ramakrishnan. Nonlinear pattern matching in trees. *Journal of the ACM*, 39(2):295–316, 1992.

[158] J. Reeder and R. Giegerich. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC Bioinformatics*, 5:104–104, 2004.

[159] D.C. Reis, P.B. Golgher, A.S. Silva, and A.F. Laender. Automatic Web News Extraction Using Tree Edit Distance. In *Proceedings of the 13th international conference on World Wide Web*, pages 502–511, New York, August 2004.

[160] T. Richter. A New Algorithm for the Ordered Tree Inclusion Problem. In *Proc. of* CPM'97*,* LNCS 1264, pages 150–166, 1997.

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

[162] E. Rivas and S.R. Eddy. Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. *Bioinformatics*, 16(7):583–605, 2000.

[163] E. Rivas and S.R. Eddy. Noncoding RNA gene detection using comparative sequence analysis. *BMC Bioinformatics*, 2(8), 2001.

[164] D.A. Rodionov, A.G. Vitreschak, and Gelfand MS Mironov AA. Comparative genomics of thiamin biosynthesis in procaryotes. New genes and regulatory mechanisms. *J. Comp. Biol.*, 277:48949–48959, 2002.

[165] B. Rössler, J. Zhang, and M. Höchsmann. Visual Guided Grasping and Generalization using Self-Valuing Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[166] J. Ruan, G. Stormo, and W. Zhang. An iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics*, 20:58–66, 2004.

[167] Y. Sakakibara, M. Brown, R. Hughey, I. Mian, R. Underwood, and D. Haussler. Recent methods for RNA modeling using stochastic context-free grammars. In *Proc. of* CPM'94*,* LNCS 807, pages 289–306, 1994.

[168] Y. Sakakibara, M. Brown, R.C. Underwood, I.S. Mian, and D. Haussler. Stochastic Context-Free Grammars for Modeling RNA. In *Proceedings of the 27th Annual Hawaii International Conference on System Sciences (HICSS-27)*, pages 284–294, 1994.

[169] H. Samet. Distance transform for images represented by quadtrees. *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 4(3):298–303, 1982.

[170] D. Sankhoff, J.B. Kruskal, S. Mainville, and R.J. Cedergren. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, chapter An Analysis of the general Tree-Editing Problem, pages 237–252. Addison Wesley, Reading, MA, 1983.

[171] D. Sankoff. Matching sequences under deletion/insertion constraints. In *Proc. Nat. Acad. Sci. U.S.A.*, volume 69, pages 4–6, 1974.

[172] D. Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problem. *SIAM Journal of Applied Mathematics*, pages 810–825, 1985.

[173] D. Sankoff, J.B. Kruskal, S. Mainville, and R.J. Cedergren. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, chapter Fast algorithms to determine RNA secondary structures containing multiple loops, pages 93–120. Addison Wesley, Reading, MA, 1983.

[174] J. Saraste, U. Lahtinen, and B. Goud. Localization of the small GTP-binding protein rab1p to early compartments of the secretory pathway. *J. Cell Sci.*, 108:1541–1552, 1995.

[175] R. Sayle and J. Milner-White. RasMol: Biomolecular graphics for all. *Trends in Biochemical Sciences (TIBS)*, 20(9):374, 1995.

[176] A. Sczyrba, J. Krüger, H. Mersch, S. Kurtz, and R. Giegerich. RNA-related tools on the Bielefeld Bioinformatics Server. *Nucleic Acids Res.*, 31(13):3767–3770, 2003.

[177] S.M. Selkow. The tree-to-tree editing problem. *Information Processing Letters*, 6(6):184–186, 1977.

[178] B. Shapiro and K. Zhang. Comparing multiple RNA secondary structures using tree comparisons. *Comp. Appl. Biosci.*, 6(4):309–318, 1990.

[179] B.A. Shapiro, J.V. Jr. Maizel, L.E. Lipkin, K.M. Currey, and Whitney C. Generating non-overlapping displays of nucleic acid secondary structure. *Nucleic Acids Res.*, 12:75–88, 1984.

[180] Bruce Shapiro. An algorithm for comparing multiple RNA secondary structures. *Comp. Appl. Biosci.*, 4(3):387–393, 1988.

[181] D. Shasha and K. Zhang. Fast Algorithms for the Unit Cost Editing Distance Between Trees. *Journal of Algorithms*, 11:581–621, 1990.

[182] H. Shi and P.B. Moore. The crystal structure of phenylalanine tRNA at 1.93 a resolution: a classic structure revisited. *RNA*, 6:1091–1105, 2000.

[183] S. Siebert and R. Backofen. MARNA: A Server for Multiple Alignment of RNAs. In *Proceedings of the German Conference on Bioinformatics*, pages 135–140, 2003.

[184] T.F. Smith and M.S. Waterman. Identification of Common Molecular Subsequences. *J. Mol. Biol.*, 147:195–197, 1981.

[185] G. Storz. An Expanding Universe of Noncoding RNAs. *Science*, 296(5571):1260–1263, 2002.

[186] J. Stoye, V. Moulton, and A.W.M. Dress. DCA: An Efficient Implementation of the Divide-and-Conquer Multiple Sequence Alignment Algorithm. *Comp. Appl. Biosci.*, 13(6):625–626, 1997.

[187] B. Stroustrup. *The C++ Programming Language (3rd Edition)*. Addison-Wesley, 1997.

[188] M. Sundaralingham and S. Rao. *Structure and Conformation of Nucleic Acids and Protein-Nucleic Acid Interactions*, pages 12–13. University Park Press, 1075.

[189] J.E. Tabaska, R.B. Cary, H.N. Gabow, and G. Stormo. An RNA folding method capable of identifying pseudoknots and base triple. *Bioinformatics*, 14(8):12–13, 1998.

[190] J.E. Tabaska and G.D. Stormo. Automated Alignment of RNA Sequences to Pseudoknotted Structures. In *Proc. of* ISMB'97, pages 311–318, 1997.

[191] K.C. Tai. The tree-to-tree corection problem. *Journal of the ACM*, 26:422–433, 1979.

[192] Y. Takahashi, Y. Satoh, H. Suzuki, and S. Sasaki. Recognition of largest common structural fragment among a variety of chemical structures. *Analytical Sciences*, 3:23–28, 1987.

[193] E Tanaka. A bottom-up computing method for the metric between trees defined by Tai. *Trans. IECE*, 66:660–667, 1983. (in Japanese).

[194] E Tanaka. An improved computing method for a metric between trees. *Trans. IECE*, 67:157–158, 1984. (in Japanese).

[195] E Tanaka. The metric between trees based on strongly structure preserving mapping. *Trans. IECE*, 67:722–723, 1984. (in Japanese).

[196] E Tanaka. A computing algorithm for the tree metric based on the structure preserving mapping. *Trans. IECE*, 68:317–324, 1985.

[197] E. Tanaka and K. Tanaka. A metric on trees and its computing method. *Trans. IECE*, 75:511–518, 1982. (in Japanese).

[198] E Tanaka and K. Tanaka. The tree-to-tree editing problem. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(2):221–240, 1988.

[199] H.M. Termin and S. Mizutani. RNA-dependent DNA polymerase in virions of Rous sarcoma virus. *Nature*, 226:1211–1213, 1970.

[200] J. Thompson, D. Higgins, and T. Gibson. CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, 22:4673–4690, 1994.

[201] I. Tinoco, O.C. Uhlenbeck, and M.D. Levine. Estimation of secondary structure in ribonucleic acids. *Nature*, 230:363–367, 1971.

[202] I. Jr Tinoco and C. Bustamente. How RNA folds. *J. Mol. Biol.*, 293:271–281, 1999.

[203] T. Töller. *Bioinformatik-Strategien zur Analyse regulatorischer RNA-Motive und ihre experimentelle Validierung*. PhD thesis, University of Bielefeld, Faculty of Technology, 2003. in German.

[204] A. Torsello. *Matching Hierachical Structures for Shape Recognition*. PhD thesis, University of York, 2004.

[205] A. Torsello and E.R. Hancock. Matching and Embedding through Edit-Union of Trees. In *Proceedings of the 7th European Conference on Computer Vision, LNCS 2352*, page 822. Springer-Verlag Heidelberg, 2002.

[206] A. Torsello and E.R. Hancock. Tree Edit Distance from Information Theory. In *Lecture Notes in Computer Science*, volume 2726, pages 71–82. Springer-Verlag Heidelberg, 2003.

[207] H. Touzet. Tree edit distance with gaps. *Information Processing Letters*, 85:123–129, 2003.

[208] D.K. Treiber and J.R. Williamson. Exposing kinetic traps in RNA folding. *Curr. Opin. Struct. Biol.*, 9:339–345, 1999.

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

[210] Gabriel Valiente. An Efficient Bottom-Up Distance between Trees. In *Proc. 8th Int. Symposium on String Processing and Information Retrieval*, pages 212–219. IEEE Computer Science Press, 2001.

[211] D. van Heesch. Doxigen. www.doxygen.org.

[212] T. Villa, J.A. Pleiss, and C. Guthrie. Spliceosomal snRNAs:Mg(2+)-dependent chemistry at the catalytic core? *Cell*, 109:149–152, 2002.

[213] R.A. Wagner and M.J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21:168–173, 1974.

[214] J.T.L Wang, B. Shapiro, D. Shasha, K. Zhang, and K.M. Currey. An algorithm for finding the largest common substructures of two trees. *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 20:889–895, 1998.

[215] J.T.L Wang and Zhang. Identifying consensus of trees through alignment. *Information Sciences*, 126:1–4, 2000.

[216] J.T.L. Wang and K. Zhang. Finding similar consensus between trees: an algorithm ans a distance hierarchy. *Pattern Recognition*, 34:127–137, 2001.

[217] J.T.L Wang, K. Zhang, and C-Y. Chang. Identifying approximately common substructures in trees based on a restricted edit distance. *Information Sciences*, 121:367–386, 1999.

[218] J.T.L Wang, K. Zhang, and G.-W. Chirn. Algorithms for Approximate Graph Matching. *Information Sciences*, 82:45–74, 1995.

[219] J.T.L. Wang, Kaizhong Zhang, and D. Jeong, Karpjoo. A System for Approximate Tree Matching. *IEEE Transactions on Knowledge and Data Engineering*, 6(4):559–571, 1994.

[220] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *J. Comp. Biol.*, 1(4):337–348, 1994.

[221] L. Wang and J. Zhao. Parametric alignment of ordered trees. *Bioinformatics*, 19(17):2237–2245, 2003.

[222] Z. Wang and K. Zhang. Alignment between Two RNA Structures. In *Proc. of* FOCS'01, pages 690–702, 2001.

[223] S. Washietl and I. Hofacker. Consensus Folding of Aligned Sequences as a New Measure for the Detection of Functional RNAs by Comparative Genomics. *J. Mol. Biol.*, 342(1):19–30, 2004.

[224] M.S. Waterman. Computer analysis of nucleic acid sequences. *Methods in Enzymology*, 164:765–792, 1988.

[225] M.S. Waterman, R. Arratia, and D.J. Galas. Pattern recognition in several sequences: consensus and alignment. *Bulletin of Mathematical Biology*, 46:515–527, 1984.

[226] J. Watson and F. Crick. A structure for Deoxyribose Nucleic Acid. *Nature*, 171:737, 1953.

[227] A. Waugh, P. Gendron, R. Altman, J.W. Brown, D Case, D. Gautheret, S.C. Harvey, N. Leontis, J. Westbrook, E. Westhof, M. Zuker, and F. Major. RNAML: a standard syntax for exchanging RNA information. *RNA*, 8(6):707–717, 2002.

[228] N. Wedemeyer, T. Schmitt-John, D. Evers, C. Thiel, D. Eberhard, and H. Jockusch. Conservation of the 3'-untranslated region of the Rab1a gene in amniote vertebrates: exceptional structure in marsupials and possible role for posttranscriptional regulation. *FEBS Letters*, 477:49–54, 2000.

[229] S. Winker, R. Overbeek, C.R. Woese, G.J. Olsen, and N. Pfluger. Structure detection through automated covariance search. *Comp. Appl. Biosci.*, 6:365–371, 1990.

[230] W. Winkler, S. Cohen-Chalamish, and R.R. Breaker. An mRNA structure that controls gene expression by binding FMN. *Proc. Nat. Acad. Sci. U.S.A.*, 99:15908–15913, 2002.

[231] K. Wojciech and B.A. Shapiro. Stem Trace: an interactive visual tool for comparative RNA structure analysis. *Bioinformatics*, 15(1):16–31, 1999.

[232] M.T. Wolfinger, W.A. Svrcek-Seiler, C. Flamm, I.L. Hofacker, and P.F. Stadler. Efficient computation of RNA folding dynamics. *J. Phys. A: Math. Gen.*, 37:4731–4741, 2004.

[233] http://rna.ucsc.edu/rnacenter/xrna/xrna. WWW.

[234] K. Yamamoto, N. Sakurai, and H. Yoshikura. Graphics of RNA secondary structure; towards an object-oriented algorithm. *Bioinformatics*, 3:99–103, 1987.

[235] W. Yamg. Identifying syntactic differences between two programs. *Software - Practice and Experience*, 21:7, 1991.

[236] K. Zhang. Algorithms for constrained edit distance between ordered labeled trees and related problems. *Pattern Recognition*, 28:463–474, 1995.

[237] K. Zhang. Efficient parallel algorithms for tree editing problems. In *Proc. of* CPM'96*,* LNCS 1075, pages 361–372, 1996.

[238] K. Zhang. Computing similarity between RNA secondary structures. In *Proceedings of the IEEE International Joint Symposia on Intelligence and Systems*, pages 126–132, 1998.

[239] K. Zhang. Efficient Parallel Algorithm for the Editing Distance between Ordered Trees. In *Proc. of* CPM'98*,* LNCS 1448, pages 80–90, 1998.

[240] K. Zhang and D. Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262, 1989.

[241] K. Zhang and D. Shasha. *Pattern Matching Algorithms*, chapter Tree Pattern Matching, pages 341–371. Oxford University Press, 1997.

[242] K. Zhang, L. Wang, and B. Ma. Computing similarity bewteen RNA structures. In *Proc. of* CPM'99*,* LNCS 1645, pages 281–293, 1999.

[243] M. Zuker. On Finding All Suboptimal Foldings of an RNA Molecule. *Science*, 244:48–52, 1989.

[244] M. Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res.*, 31(13):3406–3415, 2003.

[245] M. Zuker and AB. Jacobson. Using reliability information to annotate RNA secondary structures. *RNA*, 4(6):669–679, 1998.

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

[247] M. Zuker and P. Stiegler. Optimal Computer Folding of Large RNA Sequences using Thermodynamics and Auxiliary Information. *Nucleic Acids Res.*, 9:133–148, 1981.