Common Sequence Structure Properties and Stable Regions in RNA Secondary Structures

Dissertation zur Erlangung des Doktorgrades

der Fakultät für Angewandte Wissenschaften

der Albert-Ludwigs-Universität Freiburg im Breisgau

vorgelegt von

Dipl.-Math. Sven Siebert

September 2006

Dekan : Prof. Dr. Bernhard Nebel

Referenten : Prof. Dr. Rolf Backofen , Prof. Dr. Peter Stadler

Datum der Promotion : 4.12.2006

Zusammenfassung

Ribonukleotidsequenzen(RNA) sind einzelsträngige Sequenzen, die Strukturen unter Beachtung der Basenpaarregeln (A-U, C-G, G-U) ausprägen. Die Sekundärstruktur einer RNA ist definiert als eine Menge von Basenpaaren, welche die Einbettungseigenschaft erfüllt: für je zwei Basenpaare (i, j) und (h, l) mit $i < h$, gilt entweder $i < h < l < j$ oder $i < j < h < l$. Viele RNAs konservieren eine Struktur von Basenpaarinteraktionen besser als ihre eigentliche Sequenz. Dies verkompliziert die Analyse von RNAs und ist schwieriger zu handhaben als die der Protein und DNA Analyse. RNAs sind nicht nur Träger von Erbinformationen, sondern sind auch für katalytische und regulatorische Funktionen in der Zelle verantwortlich. Diese werden meist durch spezifische Sequenz- und Struktureigenschaften hervorgerufen. Als Beispiel sei hier das SECIS Element erwähnt, das eine stem-loop Struktur aufweist. Ist dieses Element in der unmittelbaren Umgebung im nicht translatierten Bereich (UTR) eines UGA Kodons vorhanden, so wird der eigentliche Translationsstop, der normalerweise vom UGA Kodon hervorgerufen wird, verhindert und dafür die Aminosäure Selenocystein eingebaut.

Die Erkennung und Beschreibung solcher Sequenz-/ Struktureigenschaften (wie z.B. SECIS Elemente) hat sich in der Vergangenheit als eine manuelle und zugleich ermüdende Arbeit herausgestellt. Hier ist eine automatische Analyse in Form eines multiplen Alignments unter Beachtung von Sequenz- und Struktureigenschaften wünschenswert, so wie es sie schon bei multiplen Sequenzalignments gibt.

Ein anderer wichtiger Aspekt ist die Erkennung von interessanten Sequenz/ Strukturregionen zwischen zwei gegebenen RNA Sekundärstrukturen. RNA Sequenzen können zwar mithilfe des Mfold-Programms in ihre energetisch günstige Konformation gefaltet werden, diese sichert aber noch nicht die tatsächlich biologisch relevante Konformation zu. Unter der Annahme, daß lokale Regionen höchstwahrscheinlich richtig gefaltet sind, so wie sie auch in vielen anderen suboptimalen Strukturen auftreten, gilt es, diese dann noch zu erkennen. Hierbei spielt neben der Sequenz und Struktur, eine dritte Eigenschaft eine große Rolle, nämlich die der thermodynamischen Stabilität einer RNA. Energetisch günstige Konformationen werden wahrscheinlicher von einer RNA angenommen als energetisch ungünstige.

Diese Arbeit hat als Ziel, Methoden und Algorithmen bezüglich der RNA Analyse zu erarbeiten. Neben dem Vergleich und Integration von bereits

vorhandenen Methoden, sind die folgenden Punkte der eigene Beitrag zur Dissertation:
Eigener Beitrag: Die Arbeit besteht aus hauptsächlich drei selbstständig entwickelten Methoden, die sowohl theoretisch als auch praktisch in Form von Programmen oder Web-Server erarbeitet und entwickelt worden. Die ersten beiden Punkte lassen sich in den Bereich der Sequenz-Struktur Analyse einordnen, d.h. die Sequenz einer RNA und die Sekundärstruktur einer RNA werden nicht unabhängig betrachtet, sondern in Kombination miteinander. Der dritte Punkt betrachtet die Stabilität aufgrund von thermodynamischen Parametern einer RNA:

1. Multiples Alignment von RNAs: *MARNA* ist ein Web-Server, der als Eingabe eine Menge von RNA Sequenzen erwartet, und diese unter Beachtung von Primärsequenzen und Sekundärstrukturen aligniert. *MARNA* beinhaltet eine Technik, die erstmals gute Ergebnisse ohne weitere Eingaben liefert.

2. Schnelle Erkennung von exakten Mustern in RNA Sekundärstrukturen: Hier entwickeln wir ein schnelles Verfahren mit Laufzeit $O(nm)$ und gleicher Speicherkomplexität zur Erkennung von Mustern zwischen zwei gegebenen Sekundärstrukturen. Die Ausgabe sind alle nichtüberlappenden Muster.

3. Thermodynamischen Stabilitäten von RNAs werden mithilfe von thermodynamischen Parametern im Nächste-Nachbar Modell berechnet. Während die mfe (minimum free energy) Struktur von Zuker in vernünftiger Zeit berechnet werden kann, haben wir eine lokale, modifizierte Version entwickelt, die stabile Teilstrukturen in gegebenen als auch in nicht gegebenen vollständigen Strukturen vorhersagt.

Aufbau: Diese Arbeit untergliedert sich in zwei Haupfelder. Das erste Feld deckt die Analyse von RNAs aufgrund von Sequenz- und Struktureigenschaften ab. Das zweite Feld betrachtet thermodynamische Stabilitäten von RNAs.

Kapitel 1 und 2 bringen dem Leser die Problemstellungen und Notationen von RNA Sekundärstrukturen nahe. Kapitel 3 gibt einen Überblick von bereits vorhandenen Distanzen zwischen zwei globalen RNA Sekundärstrukturen und entwickelt die Definition von Lokalität in RNAs, welche zur Berechnung

von lokalen Mustern zwischen zwei RNA Strukturen dient. Kapitel 4 behandelt das Problem von der Alignierung mehrerer RNA Sequenz-Strukturen. Hier wird unter anderem der *MARNA* Server und die darin enthaltende Methodik vorgestellt. Kapitel 5 deckt die Theorie der thermodynamischen Stabilität von RNAs ab. Hier werden die bereits bekannten Algorithmen zur Berechung der mfe Struktur als auch die Berechnung der Partitionsfunktion vorgestellt. Diese dienen zur Berechung und Vorhersage von stabilen Teilstrukturen. Kapitel 6 gibt eine detaillierte Analyse aller drei selbständig entwickelten Methoden. Kapitel 7 schließt mit einem Fazit ab und gibt noch praktische Hinweise zur Parametereinstellung von *MARNA*.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 General Context

Before the discovery of ribozymes, only proteins were known to have a catalytic activity. A rationale lies in the view that only proteins, with their complex three-dimensional structure and variety of side-groups, have the flexibility to create the active sites that catalyze biochemical reactions.

In 1967, Carl Woese, Francis Crick and Leslie Orgel were the first to suggest that RNA (ribonucleic acid) could act as a catalyst based upon findings that it can form complex secondary structures [Woese, 1967, Crick, 1968, Orgel, 1968]. The first ribozyme, an RNA molecule that can catalyze a chemical reaction, was discovered in the 1980s by Thomas R. Cech, who was studying RNA splicing in the ciliated protozoan *Tetrahymena thermophila* [Zaug and Cech, 1980]. This ribozyme was found in the intron of an RNA transcript and removed itself from the transcript. Simultaneously, Sidney Altman invested several years in studying the activities of ribonuclease P, an enzyme involved in the processing of tRNA molecules, e.g. [Altman, 1975, Altman et al., 1975, Bothwell et al., 1976, Stark et al., 1978, Kole and Altman, 1979]. In 1989, Tom Cech and Sidney Altman shared the Nobel Prize in Chemistry for their demonstration that RNA could act as an enzyme.

To date, several naturally occurring classes of catalytic RNA have been identified. Some known ribozymes include ribonuclease P (RNase P), Group I and Group II introns, leadzyme, hairpin ribozyme, hammerhead ribozyme, hepatitis delta virus (HDV) ribozyme and tetrahymena ribozyme.

Moreover, RNA has shaped up as an extremely versatile biomolecule,

which is also able to fold into complex three-dimensional structures and to interact with many other RNAs and proteins in diverse and yet specific ways. RNA has the ability to perform catalytic and regulatory functions and acts not only as a mediator carrying the information from the gene to the translational machinery. RNA may be involved in regulating gene expression caused by its specific sequence structure properties. The most prominent exceptions to the carrier function are transfer RNA (tRNA) and ribosomal RNA (rRNA), both of which are involved in the process of translation. Since the late 1990s, it has been widely acknowledged that other types of untranslated RNA molecules are present in many different organisms ranging from bacteria to mammals, and are affecting a large variety of processes including plasmid replication, phage development, chromosome structure, DNA transcription, RNA processing and modification, development control and others [Doherty and Doudna, 2001, Doudna and Cech, 2002]. Any RNA molecule that functions without being translated into a protein is called non-coding RNA (ncRNA).

To give a concrete example, the genetic code, which consists of a triplet of three nucleotides, encode 64 codons. 61 out of them encode 20 amino acids, the remaining 3 codons are terminators. The UGA codon which acts as a translation termination signal has in addition to its function as a stop codon a second function to signal the incorporation of the 21st amino acid selenocysteine (Sec). Selenocysteine is incorporated into nascent polypeptides in response to a UGA codon when a specific stem-loop structure, designated the SECIS (***Se****leno****c****ysteine* ***I****nsertion* ***S****equence*) element, is present in the 3' untranslated regions in mammalians. Proteins that contain a selenocysteine residue are called Selenoproteins. Recently, Kryukov et al. [2003] found 25 selenoproteins by describing the SECIS element from previous work [Fagegaltier et al., 2000, Kryukov et al., 1999, Lescure et al., 1999, Low and Berry, 1996, Rother et al., 2000] and screening it against the human genome.

The recognition and the description of such (SECIS-) motifs has turned out to be a manual task over the past years [Walczak et al., 1996]. Although these elements share a general pattern (see Figure 1.1), an alignment for the detection of common sequence structure properties with involvement of thermodynamic parameters remained tedious. A general automatic analysis of sequential, structural and thermodynamic properties among RNAs by means of computational resources is desirable. Furthermore, computed alignments may also help to identify an RNA as a SECIS element or not, i.e. how one RNA fits into the general pattern.

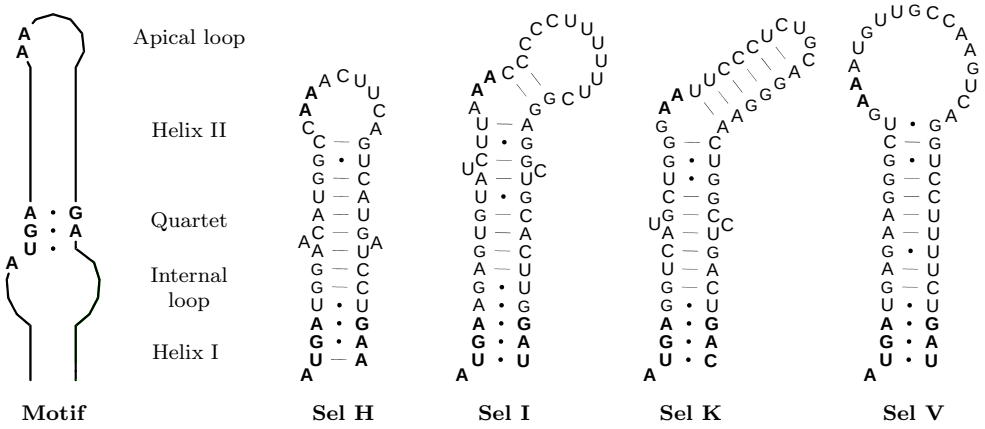


Figure 1.1: Sample set of mammalian SECIS-elements taken from Kryukov et al. [2003]. The left figure depicts the consensus structure known to be involved in all human SECIS elements. The two non-canonical A-G base-pairs are clearly visible as given in the AUGA_AA_GA pattern. The two unbounded A's in the apical loop in the motif description may occur in hairpin loops (Sel H, Sel V) or in bulged region (Sel I, Sel K).

Another aspect that is of considerable interest is to identify interesting sequence structure regions which are potentially shared by some RNA molecules. For instance, if one has a long RNA sequence which is folded by a computational program like *Mfold* [Zuker and Stiegler, 1981], then this RNA may contain an important sequence structure motif, identified e.g. as a SECIS element. The detection of such motifs, especially when considering two RNAs without predefined patterns, is the first step to describe these motifs.

Here, the assumption of a given folding structure is made. However, the assignment of a complete structure to a sequence is a bit vague. For a (long-chain) RNA there are exponentially many possible structures which may be assigned to an RNA, but assigning the correct one can only be done on the basis of a probability distribution. Even the mfe structure, which has been computed as the most stable structure, has mostly a low probabilistic value that justifies its appearance poorly. Any suboptimal structure is almost likely to be adopted as the mfe structure. Mostly, parts of RNAs are more stabilized than the surrounding nucleotides. To identify these regions may help to predict structurally stable regions of RNAs.

1.2 Sequence Structure Analysis in RNAs

Many RNAs conserve a secondary structure of base-pairing interactions more than they conserve their sequence. This makes RNA analysis more complicated and difficult than protein or DNA analysis. RNA structures can be responsible for catalytic or regulatory tasks in the cell. As we have seen in the last section, the UGA codon is responsible for the incorporation of selenocysteine in the presence of a SECIS element. The SECIS motif satisfies both sequence- and structure constraints. Thus, it is wrong to consider sequence properties or structure properties solely. A combination of both has to be taken into account. The comparison of multiple, homologous RNAs may detect similarities of sequence structure properties. They are mostly reflected in a consensus sequence and/or consensus structure. Based on this, we give a short overview of the major issues in this thesis.

Global RNA comparisons

Global RNA comparisons means to compare RNA primary sequences in combination with their secondary structures. These comparisons are accomplished by multiple alignments that allow to draw conclusions such as inferring consensus sequences/-structures. For example, the SECIS elements were aligned manually satisfying sequential and structural constraints. The alignment of the four SECIS-elements depicted in Figure 1.1 is as follows (taken from Kryukov et al. [2003]):

Conserved sequence regions as well as structural loops are aligned in an obvious manner. However, this alignment was not produced by a multiple sequence alignment. Sequence alignment programs will fail to align these

SECIS-elements correctly because of their structural features. Instead, one has to consider both the sequential and the structural properties simultaneously. There, we are faced with two subproblems. First, how to align such SECIS-elements with known structures, and second, how to align these elements with yet unknown structures.

Local RNA comparisons

Global alignment of RNAs is useful for describing similar sequence structure properties among homologous RNAs; a local variant aims at predicting similar local regions in RNAs which might be functionally important. Local alignments of DNA or protein sequences that insist on pure sequence information is solved by a fast dynamic programming approach in time $O(nm)$ and space $O(nm)$ as proposed by Smith and Waterman [1981b]. A more complicated situation arises for RNA sequences with incorporated structural constraints formed by base-pairs. Here, one has to detect similar regions in RNAs which comprises approximate sequence structure patterns between RNAs. In fact, an algorithm from Backofen and Will [2004] exists that scales $O(n^2m^2 \max(n, m))$ in time and $O(nm)$ in space. Here, n and m are the lengths of the two comparable RNA sequences. This algorithm can be used for RNAs of just moderate sizes. A fast approach to detect all exact patterns is desired to be used for long-chain RNAs. Moreover, it may be useful for assembling local, similar parts, or, in case of many agreed patterns, for aligning whole RNAs.

Thermodynamic RNA analysis

While the global and local comparison techniques deal with preferably known sequences and structures, little has been said about the probability of adopting such given structures. Predicting the correct structure of an RNA often results in computing the minimum free energy structure [Zuker and Stiegler, 1981] using thermodynamic parameters [Mathews et al., 1999] based on the nearest neighbor model. However, this method prevents from considering the abundance of suboptimal structures including an amount of important local structure properties. In particular, locally stable conformations are mostly responsible for catalytic or regulatory functions in the cell. Despite the fact that once a single, entire structure for a long-chain RNA has been computed only parts are responsible for their functions, e.g.

protein-RNA binding sites, RNA cleavage etc. These local regions crucially depend on the pre-calculated global structure; they might be undetected if they are not already contained. From the thermodynamic point of view, the frequency of base-pairs as well as the occurrence probabilities of global structures can be calculated easily. But the intermediate step to determine the frequency of any specific local structure and, moreover, to predict locally stable substructures is of particular importance in RNA structure analysis.

1.3 Objectives of this Thesis

This thesis is aimed at devising methods and algorithms in RNA structure analysis. Beside the comparison and integration of known techniques, the following points are the main contributions to this thesis. They have been devised theoretically and implemented as algorithms.

1. A multiple alignment of RNAs taking into consideration both the primary sequences and the secondary structures of RNAs, called *MARNA*.

2. A fast pattern matching algorithm that detects common pattern between two RNA secondary structures.

3. An efficient algorithm for detecting locally stable regions.

The first two points operate mainly on sequence structure properties, whereas the last point covers the theory of thermodynamic properties.

1. Multiple Alignment of RNAs has been mostly done by comparing pure nucleotide sequences as is applied for DNA sequences. Structural properties are excluded. First attempts to align RNAs allowing structural constraints are stochastic context-free grammars (SCFG) which, in turn, rely on initially good multiple alignments [Brown, 1999b]. Here, we develop a new method for aligning RNAs taking into consideration both the primary sequences and the secondary structures of RNAs. Its implementation is called *MARNA* and accessible via the *MARNA* web server at
http://www.bioinf.uni-freiburg.de/Software/MARNA or as a downloadable source code. *MARNA* is a multiple alignment method based on pairwise comparisons using a distance scoring scheme developed by Jiang et al. [2002]. *MARNA* is one of the first multiple alignment techniques for RNAs that does not rely on initial multiple alignments.

2. While *MARNA* compares complete secondary structures, no local variant of detecting sequence structure properties of RNAs exists. Recently, an algorithm that detects similar local regions in two RNA secondary structures has been published [Backofen and Will, 2004], also based on the distance scoring scheme by Jiang et al. [2002]. This algorithm suffers from being computational time-consuming. Here, we want to develop a fast pattern matching algorithm that detects exact patterns in RNA secondary structures. It is efficient in time and space and outputs not just the largest pattern but also all non-overlapping patterns.

3. Thermodynamic stability of RNAs can be computed by thermodynamic rules using the nearest neighbor model. While the mfe structure from Zuker [Zuker and Stiegler, 1981] can be computed in reasonable time, we develop a local, modified version of it to compute and predict partial, stable structures of RNA sequences.

1.4 Organization of this Thesis

This thesis is divided into two major fields. The first field is to analyze RNAs by comparing them based on their sequential and structural properties. The second field is to analyze RNAs based on their thermodynamic stability.

Chapter 2 begins with a formal description of RNAs categorizing them into primary, secondary and tertiary structures based upon structural descriptions. In this thesis, we focus on secondary structures of RNAs. Such structures are assembled from loops which, among other things, makes it possible to draw them as plots.

In chapter 3 we review different distances between two RNAs, each of them occupies a secondary structure. Here, we review algorithms computing the edit distance of RNA secondary structures, the alignment distance of RNA secondary structures under certain restrictions and a general edit distance of RNA secondary structures avoiding some restrictions. All these distances consider a pair of RNAs globally. On the other hand, local approaches are investigated by defining locality on RNAs and by considering local alignments and exact pattern matchings between two RNAs.

Chapter 4 addresses the problem of aligning multiple RNAs considering both the primary and the secondary structures. Existing methods like the Sankoff algorithm [Sankoff, 1985] and the faster approach *PMmulti* from Hofacker et al. [2004a] solve the problem of aligning and folding RNA sequences

simultaneously. They both suffer from their high computational costs. We describe *MARNA* which is a fast approach to align RNAs based on their primary sequences and on their known or unknown structures. It avoids both the high time complexity and the simplification of aligning a base-pair with either a base-pair or with two gaps as is done by the progressive profile alignment strategy proposed by Wang and Zhang [2004].

Chapter 5 covers the theory of stable RNAs in thermodynamic equilibrium. Here, a modified version of the Zuker algorithm that computes the minimum free energy of an RNA [Zuker and Stiegler, 1981] in conjunction with the partition function [McCaskill, 1990] enables to compute the occurrence probabilities of all partial structures. Two algorithms to predict the most stable substructure in a known or even unknown secondary structure are presented.

Chapter 6 gives a detailed analysis and many examples of finding exact, maximally extended patterns common to two RNA secondary structures, of *MARNA* alignments considering both the primary and secondary structures including evaluation scores and consensus- sequences and -structures and of predicting locally stable regions in RNA secondary structures.

Finally, chapter 7 draws a conclusion to all these proposed methods.

Chapter 2

Basics

Ribonucleic acid (RNA) is a nucleic acid polymer consisting of covalently bound nucleotides. Nucleotides have one, two or three phosphate groups attached to the ribose sugar. The sugar-phosphate composition constitutes the backbone of an RNA molecule. The nucleotides are linked together by the 3' carbon in the ribose of one nucleotide to the 5' carbon in the ribose of the adjacent nucleotide. A ribonucleotide is a nucleotide in which a purine or pyrimidine base is linked to a ribose molecule. The base may be adenine (A), cytosine (C), guanine (G) or uracil (U). The bases adenine and guanine belong to the purines and the bases cytosine and uracil belong to the pyrimidines. Note that thymine (T), which is found in deoxyribonucleotides(DNA), is not found as a ribonucleotide in living beings.

We describe an RNA molecule as a single sequence over the four-letter alphabet $\{\mathbf{A}, \mathbf{C}, \mathbf{G}, \mathbf{U}\}$ plus structural information. We distinguish between the primary, secondary and tertiary structures depending on the degree of structural information. In the following sections, we define these kinds of structures and give different representations of RNAs.

2.1 Primary, Secondary, Tertiary Structure of RNAs

2.1.1 Primary Structure

The easiest way to describe RNAs is by their sequential arrangement of their nucleotides. By convention, the 5' end is defined to be the starting point

and the 3' end is defined to be the end point. The sequence of nucleotides over the four-letter alphabet {**A**, **C**, **G**, **U**} is called **primary structure** (or primary sequence).

While the primary structure of a biological polymer to a large extent determines the three-dimensional shape that the molecule assumes in vivo, it mostly suffices to compare homologous sequences by aligning their primary sequences and to infer related structures. Knowing the structure of a similar sequence can completely identify the tertiary structures of the given sequences. An example of a primary structure of yeast $tRNA^{Phe}$ (Protein Data Bank (PDB), accession number 6TNA) is given as

```
5'-GCGGAUUUAG CUCAGUUGGG AGAGCGCCAG ACUGAAGAUC UGGAGGUCCU GUGUUCGAUC
   CACAGAAUUC GCACCA-3'
```

2.1.2 Secondary Structure

Most RNA molecules are single stranded that fold back onto itself to form double helical regions stabilized by the Watson-Crick base-pairs A-U and C-G and the almost thermodynamically favorable G-U base-pair. It has been observed that a base mostly participates in at most one base-pair.

Given a primary structure, i.e. a sequence S, a **secondary structure** is defined as a set $P = \{(i, j) | 1 \leq i < j \leq n\}$ of base-pairs represented as tuples of positions in the sequence of length n such that for any two base-pairs $(i_1, i_2), (j_1, j_2) \in P$ with $i_1 < j_1$ either

1. $i_1 < i_2 < j_1 < j_2$ (independence condition) or
2. $i_1 < j_1 < j_2 < i_2$ (nesting condition)

The two conditions imply that a base participates in at most one base-pair. The tuple (S, P) describes an RNA as a sequence of nucleotides provided with a secondary structure formed by base-pairs. A secondary structure can be drawn in a plane such that base-pairs are designated by arcs whose ends connect the two bonded bases, and all arcs can be drawn in one half-plane such that they do not cross. An illustration is given in section 2.3.

For many RNA molecules, the secondary structure is highly important to the correct function of the RNA, often more than the actual sequence.

The same yeast $tRNA^{Phe}$ example equipped with its classic cloverleaf structure is shown as the commonly used squiggle plot in Figure 2.1.

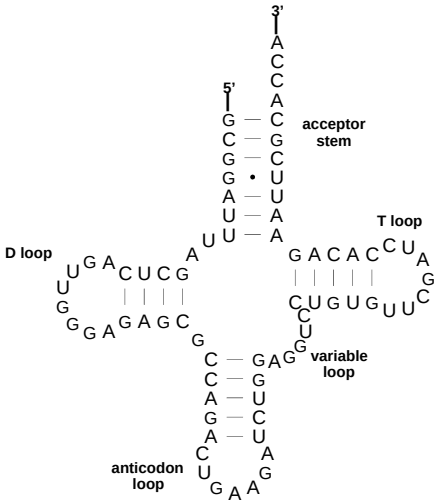


Figure 2.1: Squiggle plot of yeast $tRNA^{Phe}$.

2.1.3 Tertiary Structure

Ultimately, RNA tertiary structures are the key to understanding biological activity, and these are computationally hard to treat. **Tertiary structures** of RNAs means the spatial arrangement of secondary structure motifs in an RNA molecule or molecules. Furthermore, these structures admit base-pairing interactions which are forbidden in the secondary structures' definition. While secondary structures can be drawn as sequences with non-crossing arcs on the one half-plane, tertiary structures admit the drawings of crossing arcs. We emphasize that the definition of secondary structure excludes pseudoknots, but may occur in the tertiarys' definition. Attempts to understand large RNA tertiary structures by studying isolated secondary structural domains have met with mixed success. These constructs are readily prepared and are amenable to structure determination.

The tertiary structure of the $tRNA^{Phe}$ example shows its characteristic L-shape, representing the X-ray-crystallographic structure (see Figure 2.2). Intrinsically, this shape has emerged from its cloverleaf structure, and it is characterized through additional, unusual base interactions. These interactions consist of bindings in which three bases are involved (base-triplets). These special features are beyond our scope in this thesis.

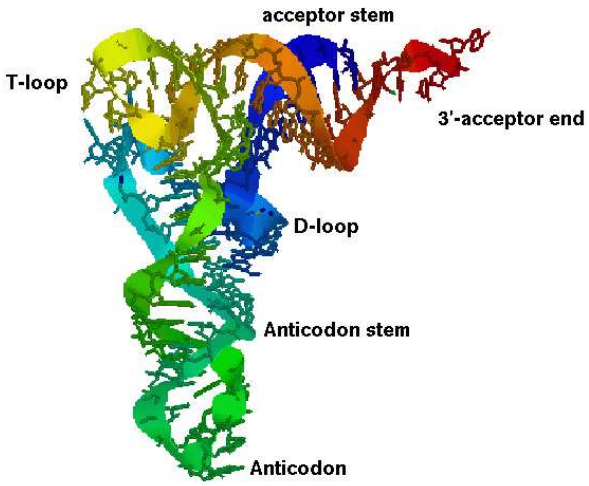


Figure 2.2: Tertiary structure of yeast $tRNA^{Phe}$.

2.2 Loop Decomposition

RNA is typically produced as a single-stranded molecule which folds back onto itself to form a number of double helical regions. For most RNA molecules, it suffices to consider secondary structures adopting sophisticated three-dimensional shapes. The base-paired structure formed by the Watson-Crick base-pairs A-U and C-G and the wobbling base-pair G-U can be divided into loops, also known as structure elements. A loop is a formation of a base-pair (i, j) that encloses a chain of nucleotides or other base-pairs. A free energy contribution in terms of practical thermodynamic parameters from the Turner lab [Mathews et al., 1999] can be assigned to each loop. The method commonly used for the energy calculation of a complete secondary structure is based on the nearest neighbor model in which the thermody-

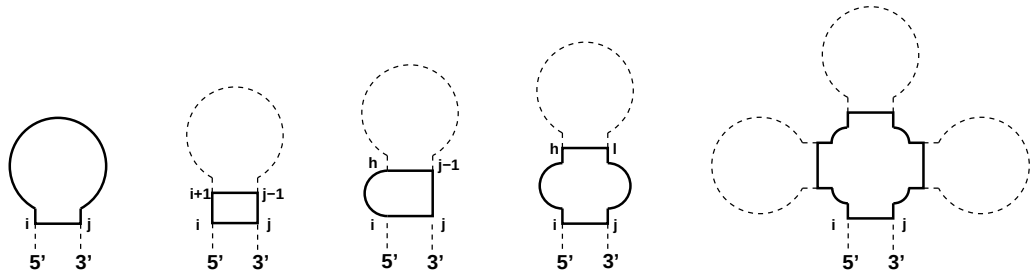


Figure 2.3: Structure elements occurring in a secondary structure.

namic stability of a base-pair is dependent on the adjacent base-pairs. The loops are assumed to contribute additively to the overall free energy of the secondary structure.

If all internal nucleotides in the sequence interval $[i+1,..,j-1]$ with base-pair (i,j) are contiguous and non-binding, then we call this element a **hairpin**. Its energy is denoted $e(hp)$. If the base-pair (i,j) is adjacent to another base-pair (h,l) such that $i < h < l < j$, then there are various structure elements: if $h > i+1$ and $j = l+1$, then we call this structure element a **left bulge**; if $h = i+1$ and $j > l+1$, then it is a **right bulge**; if $h > i+1$ and $j > l+1$, then it is an **internal loop** and if $h = i+1$ and $j = l+1$, then it is a **stack**. We summarize them by the abbreviation bis and its energy contribution by $e(bis)$. A **multi-loop** consists in addition to the base-pair (i,j) of at least two base-pairs from which several stems radiate. Its energy contribution is given as an approximation of the linear decomposition

$$e(ml) = MA + m * MB + n * MC \tag{2.1}$$

where MA, MB and MC are constants. Default values are MA=3.4, MB=0.4 and MC=0 as given in the tables of 'Free Energies at 37^0 (version 3.0)' on the webpage http://www.bioinfo.rpi.edu/ zukerm/rna/energy/ from Zuker. m is the number of base-pairs within this loop (excluding base-pair (i,j)) and n is the number of unpaired bases. All structure elements are listed in Figure 2.3.

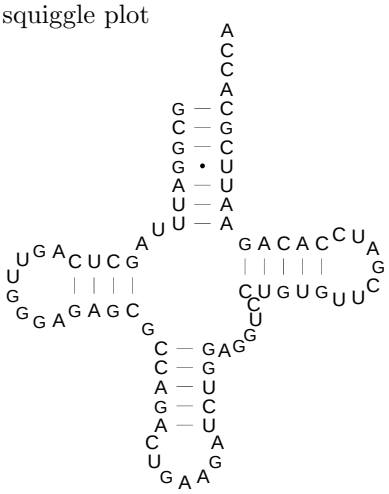
2.3 RNA Secondary Structure Representation

RNA secondary structures can be displayed in different kinds of representations. Figure 2.4 shows the most established ones. Depending on the use of the RNA molecules, specific representations are more or less useful. The bracket notation (Figure 2.4a) is a text-based representation; the structure is reflected in a string of dots and brackets. Dots denote non-bonding bases and a pair of brackets indicates a base-pair. A more convenient representation, which expands in all directions in a plane and thus is closer to a spatial representation is the squiggle plot (Figure 2.4b). It is the most prominent plot to easily describe the approximate spatial structure of an RNA. Base-pairs are given as two bases connected through either a straight line (Watson-Crick base-pairs) or a circle indicating the so-called wobbling base-pair G-U. Considering RNAs in a more theoretical way, the representations as trees or as arc-annotated sequences are well-accepted. In recent years, tree-representations of RNA secondary structures occurred in the literature, and algorithmic applications on trees are performed successfully. As an example, the distance between two trees is considered in this thesis. Arc-annotated sequences focus on representing sequences as straight lines. Arcs indicate base-pairings. This kind of representation is mainly used in this thesis due to its beneficial representation of single base and base-pair operations. A similar representation to the arc-annotated sequence is the drawing of this sequence on a circle (Figure 2.4e). Arcs are plotted as curved lines inside this circle. The mountain plot (Figure 2.4f) is useful for large RNAs. Plateaus represent unpaired regions, the heights of these mountains are determined by the number of base-pairs in which the partial sequences are embedded.

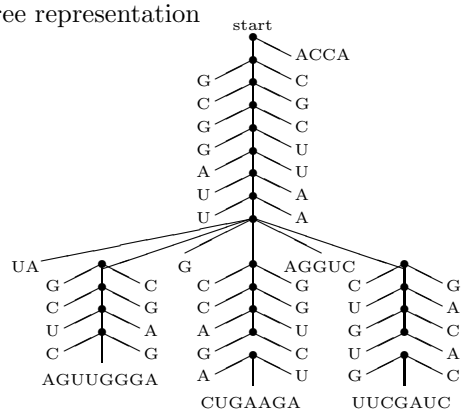
a) bracket notation

```
GCGGAUUUAGCUCAGUUGGGAGAGCGCCAGACUGAAGAUCUGGAGGUCCUGUGUUCGAUCCACAGAAUUCGCACCA
(((((((..((((........)))).(((((.......)))))).....(((((.......)))))))))))....
```

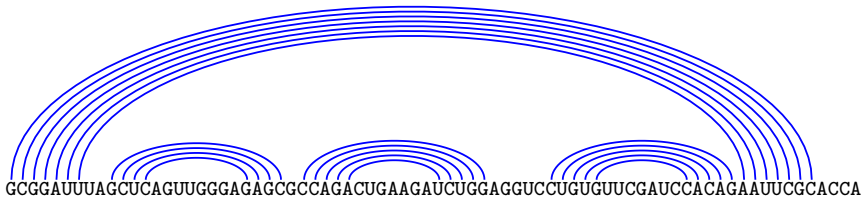
b) squiggle plot



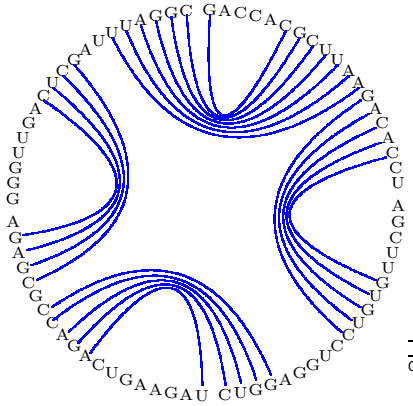
c) tree representation



d) arc-annotated sequence



e) circle representation



f) mountain plot

Figure 2.4: Different representations of the same yeast tRNA$^{\text{Phe}}$. Depending on the use of the RNA molecules, specific representations are more or less useful. The most frequent representation in this thesis is b).

Chapter 3

Pairwise Sequence Structure Comparison

A wealth of alignment algorithms dealing with sequences together with different kinds of structure information has been released in the past years. Beginning with pure sequential information, Needleman and Wunsch [1970] as well as Gotoh [1982] were one of the first who proposed sequence alignment algorithms. A local variant was developed by Smith and Waterman [1981b]. In the past years, it has been proven that structural information of RNAs are important as well. It often plays the key-role to many biological processes. It is assumed that secondary structures are often more conserved than their primary sequences. Therefore, primary sequences of RNAs cannot be aligned solely. Consequently, RNAs need to be aligned due to their structures as well. Depending on the demanding tasks, two RNAs can be aligned in the following manner:

1. two primary sequences are aligned such that both RNAs share nearly the same structure, i.e. they are folded simultaneously (see e.g. [Sankoff and Zuker, 1984, Hofacker and Stadler, 2004]),

2. one primary sequence provided with a secondary structure is aligned with another primary sequence such that the structure of the second RNA can be inferred from the first structure (see e.g. [Bafna et al., 1995, Lenhof et al., 1998, Kececioglu et al., 2000, Zhang, 1998]), and

3. both primary sequences and their structures are aligned in order to detect common sequential and structural properties (see e.g. [Jiang

et al., 2002, Zhang and Shasha, 1989, Bafna et al., 1995]).

In this chapter, we concentrate on the third case, and review existing algorithms to compare RNAs given their primary sequences and their secondary structures. The equivalence of RNA secondary structures to trees in graph theory provides a well-developed theoretical concept to compare these RNAs [Zhang and Shasha, 1989]. Based on the tree representation (see Figure 2.4c), base-pairs are assumed to be entities, i.e. a single base-pair is recognized as an entity and not just as two independent bases. We review an edit distance score between two trees from Zhang and Shasha [1989]. While the edit distance score measures the transformation from one tree into the other, the alignment score of trees measures the distance between two aligned trees. Algorithms that compute alignments of trees, and thus RNA secondary structures, are sketched [Bafna et al., 1995, Jiang et al., 2002] as well.

Local approaches are given as local alignments of RNAs which have been not yet considered extensively. Later in this chapter, we propose two methods to align RNAs locally based on their sequence structure properties. The first method is able to find all exact matchings between two RNAs in time $O(nm)$ and space $O(nm)$. Hence, this algorithm can be used for long-chain RNAs. The second method as proposed by Backofen and Will [2004] is able to find the most similar region occurring in two RNAs. Here, the time complexity is given as $O(n^2m^2 \max(n, m))$ and the space complexity is given as $O(nm)$.

3.1 Global Pairwise

3.1.1 Tree Edit Distance Applied to RNAs

Since RNA is a single strand of nucleotides that folds back onto itself into a secondary structure, the shape of this structure is topologically a tree. Internal nodes correspond to base-pairs and leaves correspond to unpaired bases in RNA secondary structures. Other representations are possible as well, e.g. each node may consist of several nucleotides that all belong to the same structure element (see also Figure 2.4c). Trees have been established in the literature due to their well-reasoned theoretical concepts and their diverse applications. RNA secondary structures can be represented as ordered labeled trees, in which the left-to-right order among siblings is significant. An algorithm that computes the distance between two ordered

labeled trees has been published by Zhang and Shasha [1989]. The distance is computed as a sequence of the weighted tree edit operations insertion, deletion and modification transforming one tree into another. Their proposed dynamic programming algorithm is capable of finding the sequence of tree edit operations with minimum costs in time $O(|T_1||T_2| \min(depth(T_1), leaves(T_1)) \min(depth(T_2), leaves(T_2)))$ and space $O(|T_1||T_2|)$, where T_1 and T_2 are trees. It is the best known algorithm to find the edit distance between two trees, and it is an improvement of the previous published algorithm from Tai [1979] with time complexity $O(|T_1||T_2|depth(T_1)^2 depth(T_2)^2)$. Here, we review the algorithm from Zhang and Shasha [1989]:

Edit Operations

We assume that we have two trees T_1 and T_2 and we want to transform one tree into another. This is done by a sequence of tree edit operations consisting of three kinds of operations:

1. *Change* a node label to another.
2. *Delete* a node such that the children of this deleted node become the children of the parent of this node.
3. *Insert* a node b by making this node a children of another node a and a consecutive sequence of siblings among the children of a become the children of b.

In fact, the insert operation is the inverse of the delete operation. An edit operation can be represented as a pair $(a, b) \neq (-, -)$, where a is either a label of a node or the gap symbol $-$ in T_1 and b is either a label of a node or the gap symbol $-$ in T_2. If both $a \neq -$ and $b \neq -$, then it is a change operation. A delete operation is given if $b = -$, and an insert operation is given if $a = -$ (see Figure 3.1).

Let S be a sequence $s_1, \ldots, s_k$ of edit operations. We call this sequence S an S-derivation from T_1 to T_2, if there exists indices $i_1, \ldots, i_k$ such that the sequence of trees $T_{i_1}, \ldots, T_{i_k}$ is derived by the edit operations s_{i_l} transforming the tree T_{i_l-1} into T_{i_l} and $T_1 = T_{i_1}$ and $T_2 = T_{i_k}$.

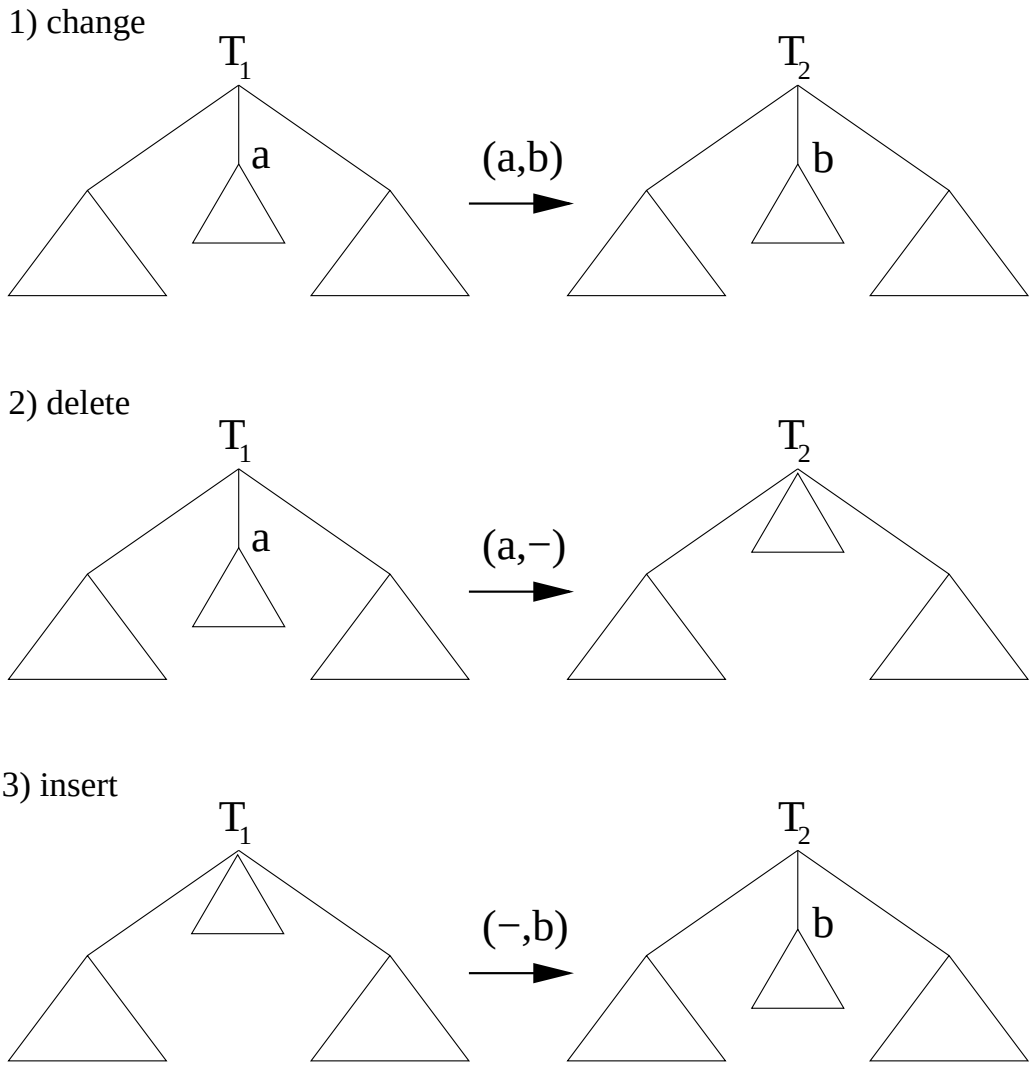


Figure 3.1: Three kinds of tree edit operations. The first edit operation is to change the label of a tree node a into b. The second operation is to delete the node with label a such that the children of a become the children of the parent of a. And the last operation is to insert a node with label b such that a consecutive sequence of children of the parent of b become the children of node b.

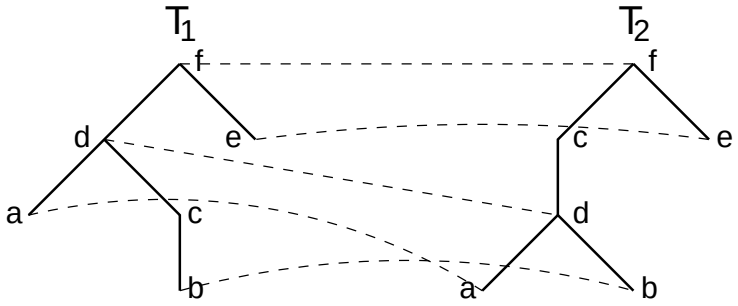


Figure 3.2: Mapping of trees.

Mapping

A tree T_1 can be transformed into another tree T_2 by a sequence of edit operations. T_1 and T_2 are two trees with N_1 and N_2 nodes, respectively. Suppose that we have an ordering for each tree, then $T[i]$ means the ith node of tree T in the given ordering. A graphical representation illustrates the transformation of trees(see Figure 3.2). A dotted line from $T_1[i]$ to $T_2[j]$ indicates the changing of node label $T_1[i]$ to node label $T_2[j]$, if $T_1[i] \neq T_2[j]$, or the node labels remain unchanged, if $T_1[i] = T_2[j]$. Any node in T_1 that is not touched by a dotted line is to be deleted. Any node in T_2 that is not touched by a dotted line is to be inserted.

Costs

To score the edit operations on trees, the γ function is used to assign a nonnegative real number to each edit operation. The cost function γ is constrained to a metric:

1. $\gamma(a, b) \geq 0; \gamma(a, a) = 0;$
2. $\gamma(a, b) = \gamma(b, a)$
3. $\gamma(a, c) \leq \gamma(a, b) + \gamma(b, c)$

If we consider again the S-derivation that transforms the tree T_1 into T_2, then the cost of S is given by $\gamma(S) = \sum_{i=1}^{|S|} \gamma(s_i)$. The optimization problem of transforming tree T_1 into T_2 with minimum costs is then given as

$$\delta(T_1, T_2) = \min\{\gamma(S) | S \text{ is a sequence of edit operations taking } T_1 \text{ to } T_2\} \tag{3.1}$$

The δ function is a metric since the γ function is a metric.

Algorithm

Let $T[i \ldots j]$ be an ordered subforest with nodes $T[i], \ldots, T[j]$ of a tree with nodes $T[1], \ldots, T[n]$. $l(i)$ is the number of the leftmost leaf descendant of the subtree rooted at $T[i]$. T is a tree with a left-to-right postorder numbering. The distance between two forests $T_1[i' \ldots i]$ and $T_2[j' \ldots j]$ is denoted $forestdist(i' \ldots i, j' \ldots j)$. The distance between the subtree rooted at i and the subtree rooted at j is denoted $treedist(i, j)$. $anc(i)$ is the set of nodes which are on the path from the root to i, inclusive.

With the help of the next definition, we are ready to specify the algorithm. Let

$$LR\_keyroots(T) = \{k | \text{ there exists no } k' > k \text{ such that } l(k) = l(k')\}$$

The set $LR\_keyroots(T)$ contains all nodes k such that k is either the root of the tree T, or, the leftmost descendant of k is not equal to the leftmost descendant of the parent of k (denoted $p(k)$), i.e. $l(k) \neq l(p(k))$. $LR\_keyroots(T)$ is an array of nodes containing them in an increasing order. This array can be computed in linear time. The algorithm to compute the distance between two trees T_1 and T_2 is then given as follows (adapted from Zhang and Shasha [1989]):

```
TREEDIST_KEYROOTS(T1, T2)
1  for i' ← 1 to |LR_keyroots(T1)|
2      do for j' ← 1 to |LR_keyroots(T2)|
3             do
4                 i = LR_keyroots[i'];
5                 j = LR_keyroots[j'];
6                 compute treedist(i, j);
```

$treedist(i, j)$
1 $forestdist(\emptyset, \emptyset) = 0$;
2 **for** $i_1 \leftarrow l(i)$ **to** i
3 **do** $forestdist(T_1[l(i) \ldots i_1], \emptyset) =$
4 $forestdist(T_1[l(i) \ldots i_1 - 1], \emptyset) + \gamma(T_1[i_1], -)$;
5 **for** $j_1 \leftarrow l(j)$ **to** j
6 **do** $forestdist(\emptyset, T_2[l(j) \ldots j_1]) =$
7 $forestdist(\emptyset, T_2[l(j) \ldots j_1 - 1]) + \gamma(-, T_2[j_1])$;
8 **for** $i_1 \leftarrow l(i)$ **to** i
9 **do for** $j_1 \leftarrow l(j)$ **to** j
10 **do if** $l(i_1) = l(i)$ *and* $l(j_1) = l(j)$
11 **then**
12 $forestdist(T_1[l(i) \ldots i_1], T_2[l(j) \ldots j_1]) = min\{$
13 $forestdist(T_1[l(i) \ldots i_1 - 1], T_2[l(j) \ldots j_1] + \gamma(T_1[i_1], -),$
14 $forestdist(T_1[l(i) \ldots i_1], T_2[l(j) \ldots j_1 - 1] + \gamma(-, T_2[j_1]),$
15 $forestdist(T_1[l(i) \ldots i_1 - 1], T_2[l(j) \ldots j_1 - 1] +$
16 $\gamma(T_1[i_1], T_2[j_1])\}$
17 $treedist(i_1, j_1) = forestdist(T_1[l(i) \ldots i_1], T_2[l(j) \ldots j_1]$
18 **else**
19 $forestdist(T_1[l(i) \ldots i_1], T_2[l(j) \ldots j_1] = min\{$
20 $forestdist(T_1[l(i) \ldots i_1 - 1], T_2[l(j) \ldots j_1] + \gamma(T_1[i_1], -),$
21 $forestdist(T_1[l(i) \ldots i_1], T_2[l(j) \ldots j_1 - 1] + \gamma(-, T_2[j_1]),$
22 $forestdist(T_1[l(i) \ldots l(i_1) - 1], T_2[l(j) \ldots l(j_1) - 1] +$
23 $treedist(i_1, j_1)\}$

Theorem 1 *The algorithm from Zhang and Shasha [1989] has time complexity* $O(|T_1||T_2| \times \min(depth(T_1), leaves(T_1)) \times \min(depth(T_2), leaves(T_2)))$ *and space complexity* $O(|T_1||T_2|)$.

The space complexity is given by the sizes of the arrays of *treedist* and $forestdist$. Each of them requires space $O(|T_1||T_2|)$. The time complexity is determined by the two for loops in the main procedure $TREEDIST\_KEYROOTS$.

The described algorithm finds the sequence of edit operations with minimum distance transforming one tree into the other. The algorithm is generalizable with the same time complexity to approximate tree matching problems. The approximate tree matching problem imply tree edit operations making the differentiation between removing a complete subtree, i.e. remov-

ing all descendants of node n (including node n), and pruning a subtree, i.e. removing all proper descendants of node n (excluding node n).

For further details, we refer to Zhang and Shasha [1989].

Remark to edit distance

The proposed algorithm by Zhang and Shasha [1989] is one of the fundamental algorithms to compute the edit distance between two trees. As already discussed in [Jiang et al., 1995], an edit distance and an alignment distance between two trees can be different. Whereas an edit transcript means to convert one tree into another by a set of pre-defined tree operations (insertion, deletion, modification), an alignment of two ordered trees T_1 and T_2 consists of inserting nodes labeled with spaces such that the resulting trees T_1' and T_2' are identical except for their labelings. The alignment distance between T_1 and T_2 is the value of an optimal alignment of T_1 and T_2. Whereas these two notions edit and alignment are different for trees, these notions are equivalent for sequences. For further readings, especially concerning the edit distance and alignment distance, we refer to Jiang et al. [1995].

Since RNA secondary structures can be represented as trees, the algorithm in the last section is useful for computing the edit distance between them, and, among other things, to identify motifs or to construct taxonomy trees [Zhang and Shasha, 1989]. The crucial point here is that, on the other hand, if one has an arbitrary alignment in terms of a sequence alignment, then it is not guaranteed to compute a score between them. The reason to this is that a given alignment need not be a tree alignment. For instance, if we consider an alignment between two sequences with secondary structures such that for base-pairs (i_1, i_2) in the first sequence and (j_1, j_2) in the second sequence position i_1 is aligned with j_1, i_2 and j_2 are both aligned with gaps, then it is not a tree alignment because these base-pairs are not recognized as a change (match or mismatch) operation, i.e. this is not an allowed tree edit operation. A more sophisticated scoring scheme has been published by Jiang et al. [2002] that is introduced in the next two subsections.

3.1.2 Simple Alignment Distance Based on Aligning Base-Pairs as a Whole

Bafna et al. [1995] and Jiang et al. [1995] propose an algorithm to align two RNA secondary structures with a scoring scheme that assumes base-pairs to

be considered as entities and to align them as a whole or not. The algorithm from Bafna et al. [1995], which is sketched here, needs time $O(n^2m^2)$ and space $O(n^2m^2)$. It has a worse time/space complexity than the algorithm given by Zhang and Shasha [1989] (section 3.1.1). Nevertheless, this algorithm provides a simple recursion equation including a commonly used comparison technique in computational RNA structure analysis.

Following the notations given in section 2.1.2, an RNA is defined as a sequence S and a list of base-pairs P satisfying the independence and nestedness condition. A base h is *accessible* from a base-pair (i, j), if $i < h < j$, and if there is no base-pair $(k, l) \in P$ such that $i < k < h < l < j$. We call the base-pair $(i, j) \in P$ the parent of (k, l), if both k and l are accessible from (i, j). Each base and each base-pair has at most one parent. The algorithm given by Bafna et al. [1995] is as follows:

Auxiliary Functions

Suppose we are given two sequences S_1 and S_2 with lengths n and m, respectively. $S_i[j]$ denotes the base at position j in sequence S_i where $j \in \{1, \ldots, |S_i|\}$. Let $S_1[0] = -$ and $S_2[0] = -$. Let A be a $2 \times n'$ matrix, such that this matrix describes the alignment of the sequences S_1 and S_2. Each row contains a string, which is interspersed with gaps. For each column j, we have either $A[1, j] \neq -$ or $A[2, j] \neq -$. Furthermore, we introduce the *gap* function:

$$gap[i, j] = \begin{cases} j & \text{, if } A[i, j] = - \\ |l < j \text{ s.t. } A[i, l] = -| & \text{, otherwise} \end{cases} \tag{3.2}$$

The *gap* function is the number of gaps that were inserted in the string S_i till the specific position j in the alignment A assuming $A[i, j] \neq -$. From the alignment A, we can read out the edit operations: if $A[1, i] = -$ and $A[2, i] \neq -$, we have an *insertion*, if $A[1, i] \neq -$ and $A[2, i] = -$, we have a *deletion* and if both $A[1, i] \neq -$ and $A[2, i] \neq -$, then it is a *base mismatch*. Furthermore, we detect a base-pair at positions i_1 and i_2 in the alignment A, if $(i - gap[1, i], j - gap[1, j]) \in P_1$ and $(i - gap[2, i], j - gap[2, j]) \in P_2$. Based on this, we introduce two functions to score base alignments

$$\gamma(u, v) = \text{score of aligning } u \text{ with } v \text{ , s.t. } u, v \in \{A, C, G, U, -\} \tag{3.3}$$

and base-pair alignments:

$$\delta(i_1, i_2, j_1, j_2) = \text{score of aligning base-pair } (i_1, i_2) \in P_1 \text{ with base-pair } (j_1, j_2) \in P_2 \tag{3.4}$$

Now, we are ready to formulate the problem in terms of the two functions and the proposed scoring scheme. For two RNAs (S_1, P_1) and (S_2, P_2) given by their sequences and structures, find an alignment that maximizes

$$\sum_{1 \le i \le m+n} \gamma(S_1[i - gap[1,i]], S_2[i - gap[2,i]]) + \sum_{1 \le i_1 < i_2 \le m+n} \delta(i_1 - gap[1,i_1], i_2 - gap[1,i_2], i_1 - gap[2,i_1], i_2 - gap[2,i_2]) \tag{3.5}$$

Algorithm

The algorithm to solve the alignment problem is then given as:

ALIGN-RNAS()

1 **for** intervals (i_1, i_2) and (j_1, j_2)

2 **do** $Align[i_1, i_2, j_1, j_2] =$

3 $$max \begin{cases} Align[i_1, i_2 - 1, j_1, j_2] + \gamma(S_1[i_2], -), \\ Align[i_1, i_2, j_1, j_2 - 1] + \gamma(-, S_2[j_2]), \\ Align[i_1, i_2 - 1, j_1, j_2 - 1] + \gamma(S_1[i_2], S_2[j_2]), \\ Align[i_1, k_1 - 1, j_1, k_2 - 1] + \\ \quad Align[k_1 + 1, i_2 - 1, k_2 + 2, j_2 - 1] + \\ \quad \delta(k_1, i_2, k_2, j_2) + \gamma(S_1[k_1], S_2[k_2]) + \gamma(S_1[i_2], S_2[j_2]), \text{ if} \\ \quad (k_1, i_2) \in P_1 \text{ and } (k_2, j_2) \in P_2 \end{cases}$$

Theorem 2 *Using the scoring scheme given above, the algorithm* ALIGN-RNAS *from Bafna et al. [1995] computes an optimal global alignment in time* $O(n^2m^2)$ *and space* $O(n^2m^2)$.

To see the time- and space- complexity from the algorithm is easy. This algorithm is based on a rather intuitive kind of view, but contains an important recurrence equation which is also used and further developed in the next section. The proposed algorithm assumes that the secondary structures of both RNAs are given, whereas Sankoff [1985] developed an algorithm to

simultaneously predicting and aligning two RNA sequences. Sankoff's algorithm carefully models the energy functions for different kinds of loops in the structure. The running time of his algorithm for two sequences is two orders of magnitude higher.

3.1.3 General Edit Distance of RNAs

Jiang et al. [2002] propose a dynamic programming approach to measure the distance between two RNAs, where one of them has a nested and the other one has a crossing structure. Here, we focus on two nested RNA structures to compute the edit distance between them. This algorithm comes along with a notion of edit operations based on single bases and on base-pairs. A base-pair is treated as a basic unit as it is already the case in the previous sections. In contrast to the previous methods (sections 3.1.2 and 3.1.1), a base-pair can be aligned with single bases and/or gaps. Hence, it can score all possible alignments. We speak of a general edit distance of RNAs.

Following Jiang et al. [2002] and the notations given in section 2.1.2, an RNA is defined as a sequence S and a list of base-pairs P satisfying the independence and nestedness condition. In accordance to the notations given in Jiang et al. [2002], we also speak of arcs instead of base-pairs, meaning that two bases (i, j) are connected by an arc. Here, we can imagine that an RNA sequence is drawn on a straight line and the two bases form a base-pair. These arc-annotated sequences are more reflecting the sequence alignments based on sequential and structural edit operations in order to distinguish them from the edit operations on trees. Furthermore, they are useful for describing recursion equations with its abundance of indices. An arc-annotated sequence, denoted (S, P), is said to be plain if there are no arcs at all, i.e. if only the primary sequence is considered. The terms *nested/crossing* arc-annotated sequences are used to represent secondary/tertiary RNA structures.

Edit Operations

Consider two arc-annotated sequences (S_1, P_1) and (S_2, P_2) and a specific alignment M. The edit distance of two nested RNAs is the minimum cost of an edit transcript that transforms one RNA into the other and vice versa. An edit transcript describes a series of edit operations performed on free bases and on base-pairs. A base $S[i_1]$ is said *free* if there is no arc incident on it. We distinguish between edit operations as specified by M performed on arcs

and on its incident bases, *and* on edit operations performed on bases which are not incident to any arc.

Edit operations on bases: Edit operations on free bases are *base match*, *base mismatch* and *base deletion*. They are already known from standard sequence alignments, e.g. Smith-Waterman algorithm, where one transforms one sequence into another while allowing operations on single bases. A base match is given if $S_1[i]$ is aligned with $S_2[j]$ and both bases are equal, i.e. $S_1[i] = S_2[j]$. If, however, $S_1[i] \neq S_2[j]$, then this operation is a base-mismatch. If $S_1[i]$ is aligned with a gap, then it is an insertion. If $S_2[j]$ is aligned with a gap, then it is a deletion. In fact, it is a base insertion operation applied to the second sequence.

Edit operations on arcs and its incident bases: For arcs, there is a more complex scoring scheme. Consider two arcs $(i_1, i_2) \in P_1$ and $(j_1, j_2) \in P_2$ such that i_1 is aligned with j_1 and i_2 is aligned with j_2. An *arc match* occurs if $S_1[i_1] = S_2[j_1]$ and $S_1[i_2] = S_2[j_2]$. We have an *arc mismatch* if $S_1[i_1] \neq S_2[j_1]$ or $S_1[i_2] \neq S_2[j_2]$. If i_1 is aligned with j_1 and i_2 is aligned with j_2 such that $(i_1, i_2) \in P_1$, but $(j_1, j_2) \notin P_2$, then we have an *arc breaking*. If exactly one of $S_1[i_1]$ and $S_1[i_2]$ is aligned with a gap, such that an arc is broken and one base is aligned with another base while the other base is aligned with a gap, then we call it an *arc altering*. If the arc (i_1, i_2) is broken and the two bases $S_1[i_1]$ and $S_1[i_2]$ are aligned with a gap, then we have an *arc removing*. The arc removing operation reflects the disappearance of the base-pair during the evolution. The last three arc operations, i.e. arc breaking, arc altering and arc removing, have the breaking of an arc in common. We summarize these operations in an *arc deletion* operation. The key idea to determine the costs of a specific alignment M is to consider the operations performed on arcs first and then on bases. For instance, if we have an arc altering, such that the arc $(i_1, i_2) \in P_1$ is aligned in such a way that i_1 is aligned with $j_1 \in S_2$ and i_2 is aligned with a gap, then we have an arc altering operation plus two edit operations on bases, namely the base (mis-)match operation, where i_1 is mapped on j_1 and the base deletion where the base i_2 is aligned with a gap. Edit operations on arcs and on bases are depicted in Figure 3.3.

Costs

We are using a distance based scoring scheme. Thus, a base match costs nothing. A base mismatch has cost w_m and a base deletion has cost w_d. Depending on the two arcs involved, an arc mismatch has cost $\frac{w_{am}}{2}$ or w_{am}.

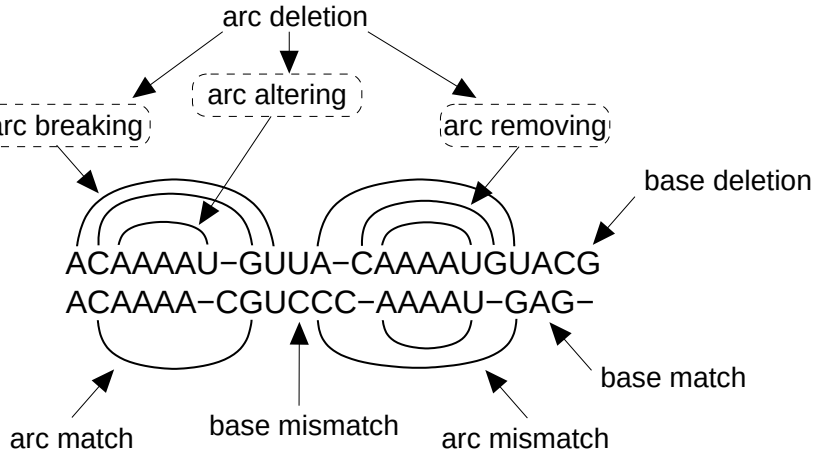


Figure 3.3: Sequence alignment with corresponding edit operations on arcs and on bases. All edit operations where one arc occurs in one sequence but disappears in the other sequence are summarized in an arc deletion operation. Note that an arc operation may be followed by a base operation.

Suppose that an arc $(i_1, i_2) \in P_1$ is aligned with an arc $(j_1, j_2) \in P_2$, then $S_1[i_1]$ is aligned with $S_2[j_1]$ and $S_1[i_2]$ is aligned with $S_2[j_2]$. If exactly one of the inequalities $S_1[i_1] \neq S_2[j_1]$ and $S_1[i_2] \neq S_2[j_2]$ holds, then the cost is $\frac{w_{am}}{2}$. If both inequalities hold, then the cost is w_{am}. An arc breaking, an arc altering and an arc removing have costs w_b, w_a and w_r, respectively. Note that for an alignment M of two RNAs, there exists arcs $(i_1, i_2) \in P_1$ and $(j_1, j_2) \in P_2$ such that $S_1[i_1]$ and $S_2[j_1]$ are each aligned with a gap, and $S_1[i_2]$ is aligned with $S_2[j_2]$, but $S_1[i_2] \neq S_2[j_2]$. Then the cost of this scenario is given as $2w_a + w_m$, because we have two arc altering operations and a base mismatch operation. Operations on arcs are performed first and then on bases. The total score of an alignment is the sum of costs of all applied edit operations.

Jiang et al. [2002] propose an algorithm for computing an alignment between two RNAs, one of them has a crossing and the other one has a nesting structure, in time $O(n^3m)$ and space $O(n^2m)$. Since we are interested in comparing two nested RNAs, the time and space complexity also holds for this particular situation. In the same paper, the authors propose an algorithm to solve the alignment problem in time $O(n^2m^2)$ and space $O(nm)$,

i.e. the improvement is in the space complexity. A necessary prerequisite for these proposed algorithms is to have a scoring scheme that satisfies the equation $2w_a = w_b + w_r$. More details to this are given in the algorithm section.

The main idea of the improvement is to distribute edit costs of arcs independently onto its incident bases except for arc matches and arc mismatches. Jiang et al. [2002] introduced two additional functions to simplify the writing of the recursion equations:

$$\psi_s(i) = \begin{cases} 1, & \text{if base not free} \\ 0, & \text{otherwise} \end{cases} \tag{3.6}$$

$$\chi(i,j) = \begin{cases} 1, & \text{if base mismatch} \\ 0, & \text{otherwise} \end{cases} \tag{3.7}$$

where s is sequence one or two and i and j are the positions of the bases in these sequences. By means of these functions and the idea of splitting costs equally onto the involved bases in arc operations, a deletion of a base $S[i]$ can now be formulated as $w_d + \psi_1(i)(\frac{w_r}{2} - w_d)$. The same idea is carried to the base match and base mismatch operations. For the base match, if both bases are free, then the cost is zero. If exactly one of the bases is free, then the cost is $\frac{w_b}{2}$. If both bases are not free, then the cost is doubled, i.e. w_b. For the base mismatch, if both bases are free, then the cost is w_m. If exactly one of the bases is free, then the cost is $w_m + \frac{w_b}{2}$. And if both bases are not free, then the cost is $w_m + 2\frac{w_b}{2} = w_m + w_b$. The last cases can be combined for the two bases $S_1[i]$ and $S_2[j]$ into the single expression $\chi(i,j)w_m + (\psi_1(i) + \psi_2(j))\frac{w_b}{2}$. The edit operations arc altering, arc breaking and arc removing are not considered explicitely, because these costs have been incorporated into the base deletion, base match and base mismatch operation.

Algorithm

With the last simplified scoring scheme, the algorithm is then given as follows:
For any $1 \leq i_1 \leq i_2 \leq n$ and $1 \leq j_1 \leq j_2 \leq m$, let

$$DP(i_1, i_2, j_1, j_2) =$$

$$\min \begin{cases} DP(i_1, i_2 - 1, j_1, j_2) + w_d + \psi_1(i_2)(\frac{w_r}{2} - w_d) \\ DP(i_1, i_2, j_1, j_2 - 1) + w_d + \psi_2(j_2)(\frac{w_r}{2} - w_d) \\ DP(i_1, i_2 - 1, j_1, j_2 - 1) + \chi(i_2, j_2)w_m + (\psi_1(i_2) + \psi_2(j_2))\frac{w_b}{2} \\ DP(i_1, k - 1, j_1, l - 1) + DP(k + 1, i_2 - 1, l + 1, j_2 - 1) \\ \quad +(\chi(k, l) + \chi(i_2, j_2))\frac{w_{am}}{2}, \\ \quad \text{if } i_1 \leq k, j_1 \leq l, (k, i_2) \in P_1 \text{ and } (l, j_2) \in P_2. \end{cases}$$

The algorithm needs time $O(n^2m^2)$ for its computation; it also seems to require a space complexity of $O(n^2m^2)$. But since one needs to maintain $DP(i + 1, i' - 1, j + 1, j' - 1)$ when $(i, i') \in P_1$ and $(j, j') \in P_2$, the space complexity is only $O(nm)$. An algorithmic rewriting of the formulas above clarifies the required space:

ALIGN-RNAS()

1 **for** $a_1 = (i_1, i_2) \in P_1$ and $a_2 = (j_1, j_2) \in P_2$
2 **do for** $i \leftarrow i_1 + 1$ **to** $i_2 - 1$
3 **do for** $j \leftarrow j_1 + 1$ **to** $j_2 - 1$
4 **do**
5 $$M(i, j) = min \begin{cases} M(i - 1, j) + w_d + \psi_1(i)(\frac{w_r}{2} - w_d), \\ M(i, j - 1) + w_d + \psi_2(j)(\frac{w_r}{2} - w_d), \\ M(i - 1, j - 1) + \chi(i, j)w_m + (\psi_1(i) \\ \qquad +\psi_2(j))\frac{w_b}{2}, \\ M(i' - 1, j' - 1) + B(a_k, a_l) \\ \qquad +(\chi(i', j') + \chi(i, j))\frac{w_{am}}{2}, \\ \qquad \text{if } a_k = (i', i) \in P_1 \text{ and } a_l = (j', j) \in P_2 \end{cases}$$
6 $B(a_1, a_2) = M(i_2 - 1, j_2 - 1)$

Here, we need two two-dimensional matrices, both not exceeding the size of nm. The matrix B contains the minimum cost of aligning the intervals $(i_1 + 1, i_2 - 1)$ and $(j_1 + 1, j_2 - 1)$ for arcs $a_k = (i_1, i_2) \in P_1$ and $a_l = (j_1, j_2) \in P_2$ provided that both arcs are aligned; i.e. we have an arc match or arc mismatch. The matrix M is constructed when considering the two arcs a_k and a_l. One matrix entry is computed in almost the same manner as a sequence alignment except that arc breaking costs are considered and computed at each single base. The algorithm proceeds from inside to outside, thereby taking arcs with minimal sequence lengths first. From the above algorithm it is easy to see that the time complexity of $O(n^2m^2)$ results from running over the arcs in both sequences and computing the best alignments

in between. The space complexity is given by the two matrices B and M. Therefore, to put the results in one sentence, we have the following

Theorem 3 *Under any score scheme satisfying $2w_a = w_b + w_r$ the problem of aligning two nested RNAs [Jiang et al., 2002] is solvable in $O(n^2m^2)$ time and $O(nm)$ space.*

The restriction to $2w_a = w_b + w_r$ in the scoring scheme is necessary to split the arc breaking costs equally on its incident bases.

A reasonable value assignment with regard to the parameters of the scoring scheme $(w_r, w_a, w_b, w_{am}, w_d, w_m)$ is given by $(2, 1.75, 1.5, 1.8, 1, 1)$ (as suggest by [Jiang et al., 2002]).

3.2 Local Pairwise

Research on sequential and structural features are associated with considerable efforts. As we have seen in the last section, sequence structure properties of RNAs are figured out due to comparison of entire RNAs.

One further important challenge is to concentrate on local regions of RNAs, i.e. to find (nearly) common patterns in RNAs. Common patterns between two RNAs are defined to share the same local sequential and structural properties. The motivation of detecting these regions is given by the fact that RNAs do not necessarily share global sequential and structural properties. This might happen if RNAs fold into different structures but share some local, stable regions.

We propose two algorithms to compute common patterns between two RNAs. The first one is able to compute all exact patterns in time $O(nm)$ and space $O(nm)$ between two RNAs. The second one is able to compute the most similar, local pattern, i.e. with gaps inside, in time $O(n^2m^2 \max(m, n))$ and space $O(nm)$ [Backofen and Will, 2004]. First of all, we give a definition of locality.

3.2.1 Locality Definition

Since the meaning of locality is more intricate in the context of sequence structure alignment than of pure sequence alignment, we explain our notion in analogy to sequence alignment. A local alignment of sequences is commonly defined as a (global) alignment of one pair of subsequences of the

input sequences. Note that the bases in a subsequence are connected via the backbone, which constitutes a dependency. For RNA, several definitions of local alignment are possible. If we define the local alignment again as the best alignment of subsequences, we ignore the RNA structure completely. Hence, in a next step we require that the subsequences represent complete substructures. This kind of locality is required for an appropriate definition of local sequence structure alignment. Additionally, one can exclude certain substructures from a substructure, while the spatial locality is preserved due to connection of bases by non-atomic H-bonds. Both the exact and the approximate algorithms are based on the atomic connection model that serve as a definition of locality, whereas the approximate algorithm additionally considers connected substructures with excluded substructures. The small example in Figure 3.4 shows that this indeed is the preferable notion of locality. The figure shows the putative SECIS-elements in the archaea *Methanococcus jannaschii* proposed by Wilting et al. [1997] (see Figure 3.4, where the putative motif is boxed). Since the apical subsequence `AAUAUAAAAUAAUAC` in the left RNA *fdhA* has no correspondence in *fwdB*, a correct local alignment of the two RNAs aligns two pairs of subsequences, which are isolated on the sequence level but connected by structure.

This should not be confused with the output of local sequence alignment programs such as BLAST [Altschul et al., 1990], which typically yield several isolated pairs of aligned subsequences. In sequence alignment, these subsequence pairs can be aligned and scored independently and are just the k best non-overlapping local alignments. However, for sequence structure alignment, the dependency created by the arcs forbids this independent treatment.

Other definitions of a local alignment of RNAs have been considered e.g. by Gorodkin et al. [2001]. They have identified common stem loops. Improving this significantly, Höchsmann et al. [2003], who handle RNA alignment by tree alignment based on an earlier work of Jiang et al. [1995], examine the problem of finding the most similar subtrees. For example, both algorithms cannot identify the motif of the RNAs in Figure 3.4 as given in the literature, since in contrast to our approach interior partial substructures are not considered. The tree alignment approach imposes some restrictions on the sequences of edit operations (and thus the alignments). In case of local alignments, the scoring scheme from Jiang et al. [2002] (see also Section 3.1.3) that constitutes the basis for general similarity overcomes these difficulties. A review of the local sequence structure algorithm (lssa) from

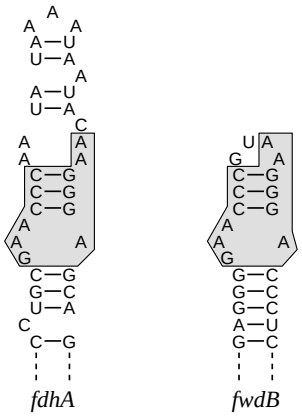


Figure 3.4: Putative SECIS-motif in non-coding regions of *Methanococcus jannaschii*(see Wilting et al. [1997]. The identical bases, which form the minimal local motif, are highlighted.

[Backofen and Will, 2004] is given with time complexity $O(n^2m^2\max(m,n))$ and space complexity $O(nm)$. First, we devise an efficient algorithm that is capable of determining all exact, local, non-overlapping patterns in time $O(nm)$ and space $O(nm)$ for two RNA molecules with lengths n and m.

3.2.2 Exact Pattern Matching

Common patterns between two RNAs are defined to share the same local sequential and structural properties. The locality is based on the connections of nucleotides given by their phosphodiester and hydrogen bonds. The idea of interpreting secondary structures as chains of structure elements leads us to develop an efficient programming approach in time $O(nm)$ and space $O(nm)$, where n and m are the lengths of the RNAs.

The patterns are maximally extended, i.e. each of them has the largest size such that no pattern completely includes the maximally extended one. As in sequence alignments, suboptimal patterns are not considered. For the patterns it means that, first, the output of proper sub-patterns is omitted, and, second, if there are at least two possibilities to build a maximally extended pattern, then these possibilities are split into a maximally extended and into a disjoint pattern. An example of the latter case follows.

The key idea of our programming method is to describe secondary struc-

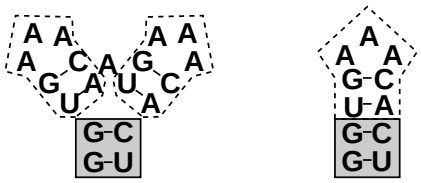


Figure 3.5: Example of extending a common pattern among two RNAs. There is no clear decision on whether to match the upper part of the stem-loop of the right RNA to the left side or to the right side of the multi-branched loop in the left RNA.

tures not only as base-pairing interactions but also as structure elements known as hairpin loops, stacks, right bulges, left bulges, internal loops or multi-branched loops (see section 2.2). By performing computations on these loop regions from inside to outside, we obtain an elegant solution to the pattern finding problem. This step is made possible because base-pairs which enclose loops occur in a nested fashion, i.e. nested base-pairs fulfill for any two base-pairs (i_1, i_2) and (j_1, j_2) either $i_1 < i_2 < j_1 < j_2$ or $i_1 < j_1 < j_2 < i_2$.

Naive Attempt

A naive attempt is to consider all combinations of positions i in the first RNA and positions j in the second RNA and to extend these starting patterns by taking into account all neighboring nucleotides in both RNAs. The neighborhood of a nucleotide is defined as the set of nucleotides which are immediately connected through their phosphodiester or hydrogen bonds. If the considered nucleotides in the neighborhood of each RNA share the same sequential and structural properties then they are taken into the pattern, and the size of this pattern is increased by one. At first glance, this idea may work, but the crucial point are the loops. Consider e.g. the case depicted in Figure 3.5: suppose the algorithm starts at position 1 (lower left corner) in the first RNA and position 1 in the second RNA and is working towards the multi-branched loop in the first RNA. The lower stem has been successfully matched. But now there is no clear decision on whether to match the upper part of the stem-loop of the second RNA to the left side or to the right side of the multi-branched loop. This decision depends on how a common pattern is defined, of course, and on how to reach the maximally extended pattern. It

is clear that the only solution here is to make some pre-computations on sequential and structural components of RNAs. Finally, we obtain a algorithm which compares inner parts of RNAs first, stores the results in various matrices and builds up the solutions successively. Note, that it is also a mistake to find common sequential parts first and then to recompose them by their structural properties. This requires exponential time since all combinations of subsets of sequence parts have to be considered.

The detection of common patterns between two RNAs can be used as a global alignment. A selection of non-overlapping patterns can be arranged in such a way that a global alignment consists of exact sequence structure parts disrupted by non-exact parts. The main application of this algorithm here is to detect large local sequence structure parts in order to find interesting regions of biologically functional similarities. Note that the worst running time is only quadratic.

Definitions and Notations

A higher level of structure interpretation is given by a classification of structure elements; we call them loops. A loop is enclosed by a base-pair (i_1, i_2). It contains all nucleotides lying on the path starting at position $i_1 + 1$ and seeking the shortest way over phosphodiester and hydrogen bonds to position $i_2 - 1$. On the way, the bonds are taken only once, and only those nucleotides are taken that lie in the inner part of the RNA, i.e. at positions $i_1 + 1$ or higher and positions $i_2 - 1$ or less.

Definition 1 (Path) *A path in an RNA (S, P) from a nucleotide at position i to a nucleotide at position j is a sequence of positions $< p_1, p_2, ..., p_k >$ such that $p_1 = i$, $p_k = j$ and there exist phosphodiester or hydrogen bonds between $S(p_{l-1})$ and $S(p_l)$ for $l = 2, \ldots, k$.*

Definition 2 (Connected Pattern) *A connected pattern of size k in an RNA (S, P) is a set of positions $T = \{p_1, p_2, ...p_k\}$ such that for any two nucleotides $S(p_i)$ and $S(p_j)$, $p_i, p_j \in T$, there exists a path from $S(p_i)$ to $S(p_j)$, completely lying in T.*

Proposition 1 *Two connected patterns in an RNA (S, P) are given by $T_1 = \{p_{1_1}, p_{1_2}, ...p_{1_k}\}$ and $T_2 = \{p_{2_1}, p_{2_2}, ...p_{2_l}\}$. If there exists at least one position p_i such that $p_i \in T_1$ and $p_i \in T_2$, then the union of the two connected patterns is connected.*

Proof 1 *Suppose there exists a position p_i with $p_i \in T_1$ and $p_i \in T_2$. For any nucleotide at position $p_r \in T_1$, there exists a path $< p_r, \ldots, p_i >$ such that each nucleotide on the path is lying in the first connected pattern. The same also holds for the second connected pattern. Hence, there exists a path from any nucleotide at position $p_r \in T_1$ to p_i to any nucleotide at position $p_s \in T_2$.*

For the comparison of patterns between two RNAs we need the following definition:

Definition 3 (partial matching) *A partial matching of two RNAs $R_1 = (S_1, P_1)$ and $R_2 = (S_2, P_2)$ is given by a set M of pairs (i, j) such that for any i there exists exactly one j (and vice versa) with*

$$M = \{(i,j) | S_1(i) = S_2(j) \wedge STR_1(i) = STR_2(j)\}$$

The first condition in M ensures that the nucleotides are equal. The second condition consists of a function $STR_k(i)$ which returns the structure type of a nucleotide at position i in sequence k ($k = 1, 2$). We make use of three structure types of a nucleotide: single stranded (ss), i.e. this nucleotide is not involved in any base-pairing interaction, left paired (lp), i.e. the nucleotide is involved in a base-pairing and the base-pair partner is located at a higher position in this sequence, and right paired (rp), i.e. the base-pair partner has a lower position.

We are interested in finding partial matchings of maximally extended, connected patterns between two RNAs. A pattern is maximally extended if there exists no pattern which has this maximally extended pattern as a proper sub-pattern under a partial matching.

We call loops which are enclosed by base-pairs inner loops. An array of nucleotides in an inner loop is a sequence of nucleotides which are connected due to their loop positions. An example of loop positions is shown in Figure 3.6.

Algorithm

Here, we study the problem of finding maximally extended, common patterns for each combination of positions i in the first RNA and j in the second RNA. We propose an efficient algorithm applied to RNAs with at most secondary structures. The positions serve as starting points from which a pattern is maximally extendable. The key idea is to maintain three $n \times m$ matrices M^{eb}, M^{nb} and M^{loop}, where n and m are the lengths of the RNAs. The

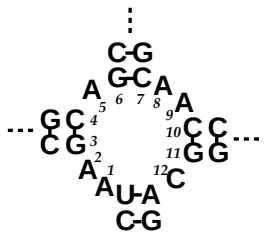


Figure 3.6: Position numbering in a multi-branched loop. The positions from 1 to 12 indicate a loop walk.

matrices correspond to calling subroutines which treat all cases of matchings: base-pair matching, only one base matching involved in a base-pairing and matching of inner loops. The algorithm works from inside to outside according to the base-pair lists. We introduce an auxiliary function *max_matching* which returns the size of a maximally extended pattern found so far in inner loops.

$bp_k(i)$ denotes the position of the base-pair partner of $S_k(i)$. The positions of an inner loop which is enclosed by a base-pair $(i, bp(i))$ is given as $< l_1, l_2, \ldots, l_{size} >$. We call the positions a loop walk according to the order the nucleotides are considered. This holds for all kinds of loops: hairpins, stacks, left/right bulges, internal loops or multi-branched loops. An example of a loop walk in a multi-branched loop is given in Figure 3.6.

Auxiliary Function: An important auxiliary function of our algorithm is *max_matching(i,j,r)*. This function is performed on inner loops starting at positions i in the first RNA and j in the second RNA with size r of a common array of nucleotides in inner loops. It returns the size of the new common pattern found so far (see Figure 3.7).

The auxiliary function makes use of matrix entries in M^{eb} and M^{nb}. They are already pre-computed due to the inside to outside step. The function is called with a parameter r which is always less than the size to the end of the current inner loop clockwise. Line 2 checks whether the nucleotides are equal. If not, then the current size is returned. If yes, then the nucleotides are checked whether they have the same structure type (line 3,5,10). If both nucleotides are non-paired(ss) then the pattern size is increased by one. Line 5 checks whether the nucleotides are left-paired. Either the base-pair partners are equal, too, then we have a base-pair matching and the pre-

```
MAX_MATCHING(i, j, r)
 1  size ← 0, m ← 0
 2  while m ≤ r and S1(i + m) = S2(j + m)
 3  do if STR1(i + m) = ss and STR2(j + m) = ss
 4        then size ← size + 1
 5     else if STR1(i + m) = lp and STR2(j + m) = lp
 6        then if S1(i + m + 1) = S2(j + m + 1)
 7                then size ← size + M^eb(pos(i + m), pos(j + m))
 8                     m ← m + 1
 9                else return size + M^nb(pos(i + m), pos(j + m))
10     else if STR1(i + m) = rp and STR2(j + m) = rp
11        then size ← size + M^nb(pos(i + m), pos(j + m))
12     m ← m + 1
13  return size
```

Figure 3.7: Auxiliary function for computing the size of a maximally extended pattern between two RNAs.

computed value is in M^{eb} (line 7), or the base-pair partners are not equal, i.e. the pre-computed value is in M^{nb} (line 9). In the latter case, we know that the connectivity of a pattern is broken; the procedure stops extending this pattern by returning the current size. The situation given in line 10 can only occur if r=0, i.e. we know that the left base-pair partners are not equal since the call of this procedure is only done to the right base-pair partners from the procedure *loop_walking*. The positions in this function operate on loop positions. The *pos* function returns global positions of RNAs.

Loop Walking: Loop walking computes the sizes of maximally extended, common patterns in inner loops. It is called from the main procedure with positions i and j in the first and second RNA, respectively. See Figure 3.8.

The values of $l_{i_{size}}$ and $l_{j_{size}}$ are the numbers of nucleotides in inner loops to the end of inner loops considering them clockwise. They can be easily obtained by traversing the inner loop in advance. The while-loop in line 6 determines the sizes of common nucleotide arrays in inner loops including structure type checking (line 8). The sizes of the new common patterns found so far will be given by the procedure *max_matching* (line 10,11). This step is only done if the tuples (k, l) are not yet considered. This means that if

```
LOOP_WALKING(i, j)
 1  l_{i_1} ← 1(= global position i)
 2  l_{j_1} ← 1(= global position j)
 3  for k ← l_{i_1} to l_{i_size}
 4  do for l ← l_{j_1} to l_{j_size}
 5       do r ← 0
 6          if (k, l) not yet considered
 7            then while k + r < l_{i_size} ∧ l + r < l_{j_size}
 8                      ∧   S_1(k + r) = S_2(l + r)
 9                      ∧   STR_1(k + r) = STR_2(l + r)
10                      do r ← r + 1
11                      M^loop(pos(l), pos(k)) ←
12                              max_matching(k, l, r)
```

Figure 3.8: *Loop walking* computes the size of a maximally extended, common patterns in an inner loop starting at position i in the first RNA and j in the second RNA.

the size of the same nucleotide array starting at positions k and l is at least 2, then the value of $M^{loop}(pos(l+1), pos(k+1))$ needs not be recomputed since it has already been considered. This information can be easily stored in a binary $n \times m$ matrix. Note again, that the procedure operates on loop positions.

Base-Pair Matching: Here, we assume that a base-pair $(i, bp(i))$ in the first RNA matches a base-pair $(j, bp(j))$ in the second RNA. We distinguish two cases of obtaining the correct size for the maximally extended pattern. The computation takes place at $M^{eb}(i, j)$. We omit the algorithmic code here.

Case 1: The maximal, common array of nucleotides in inner loops to the right of positions i and j and to the left of $bp(i)$ and $bp(j)$ do not overlap. See Figure 3.9 for an illustration.

The value of $M^{eb}(i, j)$ is given as $M^{loop}(i+1, j+1) + M^{loop}(i', j') + 2$, where i' and j' are the first positions left, i.e. counter-clockwise, to $bp(i)$ and $bp(j)$ sharing the same common array of nucleotides from i' to $bp(i) - 1$ and from j' to $bp(j) - 1$. Either the size of the array is greater than 1, then $M^{loop}(i', j') \neq 0$, otherwise we set $M^{loop}(i', j') = 0$

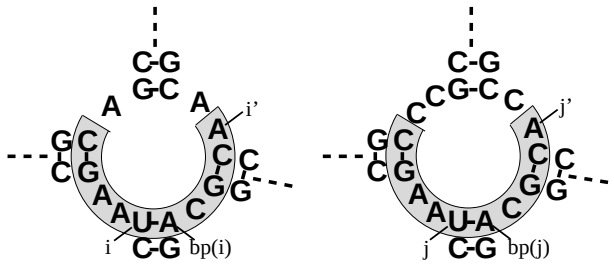


Figure 3.9: The base-pairs $(i, bp(i))$ and $(j, bp(j))$ match. The first case is given by the non-overlapping common array of nucleotides in inner loops to the right of positions i and j and to the left of bp(i) and bp(j).

in the above sum. Finally, the matrix entries of $M^{loop}(i+1, j+1)$ and $M^{loop}(i', j')$ are set to 0 since the sizes of these patterns are now included in $M^{eb}(i, j)$. This step is necessary to prevent the output of proper sub-pattern. The base-pair matching, i.e. (i, j) matches (i', j'), is counted as a pattern of size 2.

Case 2: The maximal, common array of nucleotides in inner loops to the right of positions i and j and to the left of bp(i) and bp(j) do overlap in at least one RNA. See Figure 3.10 for an illustration.

As shown in Figure 3.10, there is no clear decision how the nucleotides should be matched. In the left figure, the two A's immediately before the cutting line are able to match the rightmost A's or the leftmost A's or one leftmost and one rightmost A in the right figure. This is the only situation where the entire assignment becomes ambiguous, and hence, we get an exponential number of solutions if all matchings were considered. Fortunately, a maximum pattern can be extracted by looking first at which sub-array of nucleotides in the cycle is able to match to both sides of the other inner loop. In our example, this sub-array consists of the two A's. It can be obtained by traversing the inner loops from positions $i+1$ and $j+1$ and by matching their corresponding nucleotides until no matching is possible any more or the positions $bp(i)-1$ or $bp(j)-1$ are reached. The same procedure is performed on positions $bp(i)-1$ and $bp(j)-1$ counter-clockwise. The overlapping array of nucleotides is ambiguous to match. We proceed by going from one nucleotide to the next in the sub-array and make

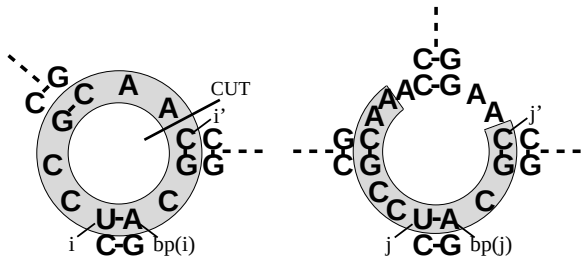


Figure 3.10: The base-pairs $(i, bp(i))$ and $(j, bp(j))$ match. The second case is given by the overlapping common array of nucleotides in inner loops to the right of positions i and j and to the left of bp(i) and bp(j).

a cut such that the right/left side of the first RNA is matched to the right/left side of the second RNA. For both sides, the *max_matching* values are computed. Since we don't want a cubic time algorithm, we add and subtract the corresponding *max_matching* values of the current nucleotides in the sub-array. This prevents having to re-compute the whole *max_matching* values. Note that base-pairs are not broken, i.e. if there is a base-pair matching in inner loops then the step is to jump over the base-pair partner. The cut which provides a maximum value is then chosen; i' and j' are the positions immediately after the cut. The value of $M^{eb}(i, j)$ is given as $M^{loop}(i+1, j+1) + M^{loop}(i', j') + 2$. As in case 1, the matrix entries of $M^{loop}(i+1, j+1)$ and $M^{loop}(i', j')$ are set to 0. Of course, both inner loops may have the overlapping situation, but it does not affect the time complexity.

Non-Base-Pair Matching: The last matching case is when only one base of a base-pairing in one RNA is matched with one base of the same structure type in the other RNA. Thus, overlapping regions can be excluded. Entries of M^{nb} can be computed uniquely. The algorithmic code is given in Figure 3.11.

Line 1 checks if the bases of structure type lp are the same. If it succeeds, then the entry of $M^{nb}(i, j)$ is computed. The same holds for the bases with structure type rp (line 4). The procedure is called, if the nucleotide at position i is involved in a left base-pairing in the first RNA and the nucleotide at position j in a left base-pairing in the second RNA. i' and j' are the leftmost positions such that $M^{loop}(i', j') \neq 0$, if both RNAs share the same common array of nucleotides in inner loops from i' to $bp(i)$ and from j'

NONE_BASE-PAIR_MATCH(i, j)
1 **if** $S_1(i) = S_2(j)$
2 **then** $M^{nb}(i, j) = M^{loop}(i + 1, j + 1) + 1$
3 $M^{loop}(i + 1, j + 1) = 0$
4 **else if** $S_1(bp(i)) = S_2(bp(j))$
5 **then** $M^{nb}(bp(i), bp(j)) = M^{loop}(i', j') + 1$
6 $M^{loop}(i', j') = 0$

Figure 3.11: Algorithm for the none base-pair matching case.

to $bp(j)$, otherwise $M^{nb}(bp(i), bp(j))$ is set to 1.

Main Procedure: The main procedure performs the pattern search from inner to outer loops. It consists of taking base-pairs $(i, bp(i))$ from the first and $(j, bp(j))$ from the second RNA. A call on *loop_walking(i+1,j+1)* is executed. Depending on the base-pair matching cases, the main procedure proceeds by calling either the *base-pair_match(i,j)* procedure or the *none-base-pair(i,j)* matching procedure. A special kind of an inner loop is the external loop starting at the first position and ending at the last position. Here, we assume that both RNAs are enclosed by a virtual base-pair in order to execute the *loop_walking* procedure, but the remaining procedures are not executed any more.

Traceback: We have stored all sizes of maximally extended, common patterns in M^{loop}. The starting positions i and j are given by entries $M^{loop}(i, j) \neq 0$. The traceback retrieves maximally extended patterns using the same cases of matching base-pairs, matching none base-pairs and matching arrays of nucleotides in inner loops. One can make use of an additional $n \times m$ matrix, which stores the sizes of cuts at the appropriate positions during computation. Then the traceback step for one pattern will be not worse than linear time.

Complexity

Time complexity : The main procedure consists of running through all combinations of base-pairs in both RNAs. This yields a running time of $O(nm)$. For each combination, the *loop_walking* procedure is executed only once. The procedure operates on each combination of nucleotide positions in inner loops. Since inner loops do not overlap each other in one RNA, this

does not increase the running time. The *loop_walking* procedure calls the *max_matching* procedure which also operates on inner loop positions. The *max_matching* procedure marks nucleotides as considered if they are matched successfully. This is noticed by the *loop_walking* procedure such that both procedures consider each combination of inner loop positions almost twice. Therefore, the running time of $O(nm)$ is retained. The *base-pair_match* procedure consists of finding two common arrays of nucleotides in both directions of a base-pair combination. Finding the cutting line means to traverse the overlapping regions. This does not cost more than the number of nucleotides in inner loops. The *non-base-pair* procedure has cost of finding a common array of nucleotides from the right base-pair partner. Therefore, we yield a running time of

$$O(nm)$$

Space complexity: The algorithm needs three $n \times m$ matrices M^{eb}, M^{nb} and M^{loop} for computing the sizes, one $n \times m$ boolean matrix for marking considered nucleotides and one $n \times m$ matrix for storing positions of cuts. The space complexity is thus given by

$$O(nm)$$

3.2.3 Approximate Matching

A local approximate matching consists of detecting similar regions between two RNAs. To distinguish this from the matching algorithm proposed in the last section, we review here a $O(n^2m^2 \max(n, m))$ time and $O(nm)$ space algorithm that allows some inaccuracies in their patterns [Backofen and Will, 2004]. It is accomplished by defining the local sequence structure alignment problem (lssa). This algorithm is based on the notion and the scoring scheme defined by Jiang et al. [2002] and introduced in section 3.1.3. Recall that Jiang et al. [2002] solved the problem of globally aligning two nested RNAs in time $O(n^2m^2)$.

Local Similarity

In the following, we fix two nested RNAs $S_1 = (S_1, P_1)$ and $S_2 = (S_2, P_2)$. An *alignment A of two sequences S_1 and S_2* is a subset of $[1..|S_1|] \cup \{-\} \times [1..|S_2|] \cup \{-\}$, where for all pairs $(i, j), (i', j') \in A$ holds

1. $i \leq i' \Rightarrow j \leq j'$,
2. $i = i' \neq - \Rightarrow j = j'$, and
3. $j = j' \neq - \Rightarrow i = i'$.

Given an arc $(i, i') \in P_1$ (resp. $(i, i') \in P_2$), then the arc (i, i') is called *aligned* in the alignment A if and only if there is an arc (j, j') in the other structure P_2 (resp. P_1), such that $(i, j), (i', j') \in A$ (resp. $(j, i), (j', i') \in A$). I is called *arc-complete for a structure* P, write $ac_P(I)$, iff for every arc $(i, i') \in P$ holds either $i, i' \in I$ or $i, i' \notin I$. The term arc-complete formalizes the treatment of an arc as an entity. By restricting ourselves to arc-complete sets, we disallow that the two bases of each arc are separated. Let $\pi_1(A)$ (resp. $\pi_2(A)$) denote the projection to the aligned positions in the first (resp. second) sequence of A. Then, we call A *arc-complete* for P_1 and P_2, iff $\pi_l(A)$ is arc-complete for P_l for $l \in \{1, 2\}$. An *exclusion of A in sequence $l = 1, 2$* is a range $[k..k']$, where $k \leq k'$, such that $[k..k'] \notin \pi_l(A)$ and $k-1, k'+1 \in \pi_l(A)$.

Definition 4 (LSSA Problem) *Let $\mathcal{S}_1 = (S_1, P_1)$ and $\mathcal{S}_2 = (S_2, P_2)$ be RNAs with nested structures. An alignment of $\mathcal{S}_1$ and $\mathcal{S}_2$ is called* local sequence structure alignment (lssa) of $\mathcal{S}_1$ and $\mathcal{S}_2$, *if and only if 1) A is arc-complete and 2) any exclusion of A has an immediate aligned successor a and no second exclusion has a as an immediate aligned successor. Further, given $\mathcal{S}_1$, $\mathcal{S}_2$, and a similarity score* SimScore, *the* lssa problem *is to determine*

$$\underset{A \text{ lssa of } \mathcal{S}_1 \text{ and } \mathcal{S}_2}{\arg\max} \textsc{SimScore}(A, \mathcal{S}_1, \mathcal{S}_2).$$

The definition of a successor and a detailed description can be read in [Backofen and Will, 2004]. Here, we are satisfied with the fact, that one has to determine the value $\arg\max \textsc{SimScore}(A, \mathcal{S}_1, \mathcal{S}_2)$

For the definition of similarity score, the functions s_b, s_{arc}, s_{br1}, and s_{br2} are introduced. $s_b(i, j)$ denotes the similarity between the bases $S_1[i]$ and $S_2[j]$. If $i = -$ (resp. $j = -$), then $s_b(i, j)$ is the similarity between $S[j]$ (resp. $S[i]$) and a gap. $s_{arc}(a_1, a_2)$ is the similarity between arcs a_1 and a_2. Moreover, $s_{br1}(a_1, j, j')$ is the penalty (i.e. a negative similarity) for breaking the arc a_1 by aligning its ends to j and j' in sequence S_2, where $(j, j') \notin P_2$, or to gaps. Analogously, $s_{br2}(i, i', a_2)$ is defined as penalty for breaking the arc a_2 in P_2.

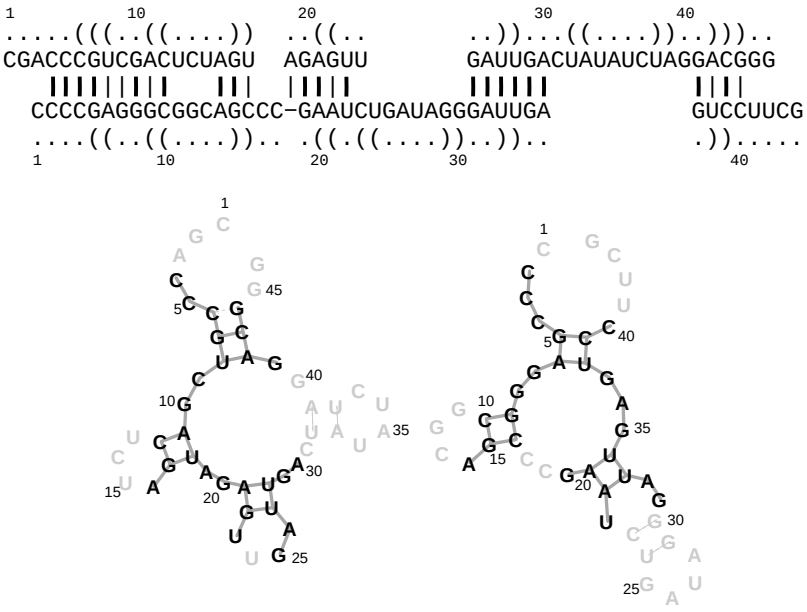


Figure 3.12: Local alignment of two RNAs. The upper part shows the two local, aligned parts of each RNA connected via vertical bonds. The lower part shows the same RNA structures as squiggle plots. Dark nodes represent aligned nucleotides and the light ink nodes represent unaligned nucleotides.

Let A be an (arbitrary) alignment of $\mathcal{S}_1$ and $\mathcal{S}_2$. The *general similarity score of A given by the functions s_b, s_{arc}, s_{br1}, and s_{br2} for $\mathcal{S}_1$ and $\mathcal{S}_2$* is defined as

$$\begin{aligned}\textsc{SimScore}(\mathcal{S}_1,\mathcal{S}_2,A) \;=\; & \sum_{\substack{(i,j)\in A\\ \neg inc_1(i)\wedge\neg inc_2(j)}} s_b(i,j) \;+\; \sum_{\substack{(i,i')\in P_1,(j,j')\in P_2\\ (i,j)\in A,(i',j')\in A}} s_{arc}((i,j),(i',j')) \\ & +\; \sum_{\substack{(i,i')\in P_1,(j,j')\notin P_2\\ (i,j)\in A,(i',j')\in A}} s_{br1}((i,i'),j,j') \;+\; \sum_{\substack{(i,i')\notin P_1,(j,j')\in P_2\\ (i,j)\in A,(i',j')\in A}} s_{br2}(i,i',(j,j')).\end{aligned} \tag{3.8}$$

See Figure 3.12 for a local sequence structure alignment example.

Algorithm

The proposed dynamic programming algorithm for the *lssa problem for two nested RNAs $\mathcal{S}_1 = (S_1, P_1)$ and $\mathcal{S}_2 = (S_2, P_2)$* is as follows. For applying

dynamic programming to this problem, several recursion equations were developed, that recursively define the maximal similarity score of a local alignment of sequences $\mathcal{S}_1$ and $\mathcal{S}_2$. These recursion equations can be efficiently evaluated while filling matrices, i.e. materializing intermediate results. Finally, the actual optimal alignment can be obtained by traceback from the matrices. We omit listing all recursion equations here, but, again, we refer to [Backofen and Will, 2004]. However, we review the case distinctions which are implied in the used matrices.

In the following, fix arcs $a_1 \in P_1$ and $a_2 \in P_2$. For sets $I \subseteq \{1, \ldots, n\}$ and $J \subseteq \{1, \ldots, m\}$, we introduce $E(I, J)$ as an abbreviation for the set of edges $I \cup \{-\} \times J \cup \{-\}$. At the center of the system of recursion equations, $D(a_1, a_2)$ is defined as the maximal similarity score of a lssa $A \subseteq E([a_1^l..a_1^r], [a_2^l..a_2^r])$ of $\mathcal{S}_1$ and $\mathcal{S}_2$, where the two arcs a_1 and a_2 match and a_1^l denotes the left arc end in the first RNA (analogously for a_1^r, a_2^l, a_2^r). The scores are materialized in a matrix D and will be used to compute the maximal score of a lssa of $\mathcal{S}_1$ and $\mathcal{S}_2$. For the recursive definition of an entry $D(a_1, a_2)$, four further recursion equations are needed.

To compute a score $D(a_1, a_2)$ we have to consider sub-sets of local alignments restricted to bases between the left and right ends of the two arcs a_1 and a_2. Since the arc-match between a_1 and a_2 justifies exclusions in the local alignments, those restrictions, which do not contain this arc-match, are not necessarily local alignments themselves. Furthermore, they need not to be arc-complete. We express this by the following definitions.

A *top-level exclusion in sequence l of alignment A* ($l = 1, 2$) is an exclusion $[k..k']$ in sequence l of A, where neither k nor k' have a successor in P_l^A. A *local sub-alignment* $A \subseteq E([i..i'], [j..j'])$ of $\mathcal{S}_1$ and $\mathcal{S}_2$ is an alignment, satisfying the conditions of a local alignment except that in each sequence one top-level exclusion is allowed and the alignment is sub-arc-complete instead of arc-complete (compare to Definition 4).

Now, we define the maximal similarity score of all local sub-alignments $A \subseteq E([a_1^l..i], [a_2^l..j])$ of $\mathcal{S}_1$ and $\mathcal{S}_2$, where $i < a_1^r$ and $j < a_2^r$, recursively going back to smaller i and j. As a particularity, we have to ensure that there is at most one top-level exclusion in each sequence.

We count the top-level exclusions by keeping track of four states. Namely, for $i < a_1^r$ and $j < a_2^r$, we define the maximal similarity score of a local sub-alignment $A \subseteq E([a_1^l..i], [a_2^l..j])$ of $\mathcal{S}_1$ and $\mathcal{S}_2$, where the sub-alignment has

1. at most one exclusion in every sequence (arbitrary local sub-alignment)

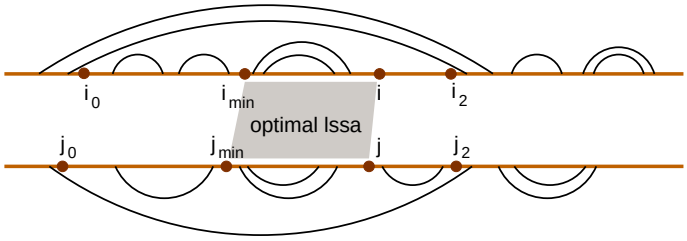


Figure 3.13: Example for the last step ("top-level") of the local alignment. The region of the optimal lssa that is scored by T is highlighted.

2. at most one exclusion in the first sequence
3. at most one exclusion in the second sequence
4. no exclusions (true local alignment)

For applying dynamic programming, four recursion equations were given, one for each case (see [Backofen and Will, 2004]). A last step is needed which computes the local alignment on the "top-level" (see Figure 3.13). It is also a recursion equation represented by the matrix T that determines the best local sequence structure alignment by searching the maximal entry in the matrix T

Altogether, we summarize the results and the need for computational resources in

Theorem 4 *For a similarity score* SimScore *and nested RNAs* $\mathcal{S}_1 = (S_1, P_1)$ *and* $\mathcal{S}_2 = (S_2, P_2)$ *with lengths* n *and* m*, respectively, there is a* $O(n^2m^2 \max(n, m))$ *time and* $O(nm)$ *space algorithm for the lssa problem [Backofen and Will, 2004].*

Chapter 4

Multiple Alignment

In the last chapter, we proposed methods to compare two RNA sequences together with their secondary structures. The results are distance measures and alignments including the inspection of base-pairs. It has been turned out that the scoring scheme as proposed by Jiang et al. [2002] is the most flexible and the most reasonable one with time complexity $O(n^2m^2)$ and space complexity $O(nm)$. In this chapter, our aim is to extend the pairwise comparison technique to a multiple comparison technique. Multiple alignments are an essential prerequisite to many further analyses such as homology modeling, motif description or illustration of conserved and variable binding sites. For RNAs, many of the alignments in the literature have been done manually, which is a tedious task. Hence, automatic methods which are fast and precise are desired. Multiple alignments are a generalization of pairwise alignments. The construction of an optimal alignment is an NP-complete problem, even for sequence alignments. In practice, some heuristic methods carried through the literature sacrificing some of the exactness but are feasible for the commonest tasks.

In this chapter, we start with the most popular sequence alignment method *ClustalW* from Thompson et al. [1994], usually applied to protein sequences. The concept is to align sequences in a progressive clustering strategy. Based on this, improvements have been developed in the *T-Coffee* system [Notredame et al., 2000] avoiding gap regions that cannot be rectified by the *ClustalW* program. A collection of weighted alignment edges, called library, produce strongly conserved regions. Albeit the program has been intended for proteins, the main idea is transferred to RNAs including sequence and structure constraints, which are realized in the *MARNA* system (chapter 4.3). In addi-

tion, we review other RNA alignments as given by *PMmulti* [Hofacker et al., 2004a] and by Wang and Zhang [2004].

4.1 Sequence Alignments (ClustalW, T-Coffee)

Definition of Multiple Alignments

Here, we assume that we want to align sequences without any structural constraints. We assume that columns of an alignment are statistically independent. A gap receives a linear gap penalty of d, such that a sequence of gaps with length g is given by $\gamma(g) = gd$. The overall score $S(m)$ for an alignment is calculated as the sum of the scores $S(m_i)$ for all columns i:

$$S(m) = \sum_i S(m_i) \tag{4.1}$$

The linear gap penalty can be replaced by the affine gap penalty. But this will make the notations more complicated. Using the affine gap penalty, we are now able to write the recurrence relation for the dynamic programming algorithm:

$$\alpha_{i_1,i_2,\ldots,i_N} = \max \begin{cases} \alpha_{i_1-1,i_2-1,\ldots,i_N-1} & +S(x^1_{i_1}, x^2_{i_2}, \ldots, x^N_{i_N}), \\ \alpha_{i_1,i_2-1,\ldots,i_N-1} & +S(-, x^2_{i_2}, \ldots, x^N_{i_N}), \\ \alpha_{i_1-1,i_2,i_3-1,\ldots,i_N-1} & +S(x^1_{i_1}, -, \ldots, x^N_{i_N}), \\ & \vdots \\ \alpha_{i_1-1,i_2-1,\ldots,i_N} & +S(x^1_{i_1}, x^2_{i_2}, \ldots, -), \\ \alpha_{i_1,i_2,i_3-1,\ldots,i_N-1} & +S(-, -, \ldots, x^N_{i_N}), \\ \alpha_{i_1,i_2-1,\ldots,i_{N-1}-1,i_N} & +S(-, x^2_{i_2}, \ldots, -), \\ & \vdots \end{cases} \tag{4.2}$$

This equation contains all combinations of gaps except the one where all residues are replaced by gaps. There are $2^N - 1$ such combinations. The algorithm requires the computation of the whole dynamic programming matrix with $L_1 L_2 \ldots L_N$ entries. Each entry is the maximum of the $2^N - 1$ combinations of gaps, excluding the case where all gaps occur in a column. If we assume that all sequences have nearly all the same sequence length

$\overline{L}$, then the time complexity is given by $O(\overline{L}^N)$ and the space complexity is given by $O(2^N\overline{L}^N)$.

Therefore, a reduction of the computational resources is needed by introducing heuristic methods. The most commonly used heuristic method is based on the progressive alignment approach from Feng and Doolittle [1987]. It has been carried out in a Clustal series, the most popular is the *ClustalW* implementation. Methods other than the progressive alignment approaches are simultaneous alignments, implemented e.g. in the package MSA [DJ et al., 1989] or DCA [Perrey et al., 1997] based on the Carrillo and Lipman (1988) algorithm. Iterative strategies [Gotoh, 1996, Notredame and Higgins, 1996] search for local optimal alignment solution, but may fail the global optimal solution. The simultaneous alignments lack in their computational speed and efficient memory exploitation.

ClustalW

The most popular representative of the progressive alignment strategy is the *ClustalW* program from Thompson et al. [1994]. In general, the progressive alignment strategy is a method of constructing a multiple alignment by successively combining sub-alignments into bigger alignments. Initially, two sequences are chosen and aligned by standard pairwise alignment.

The proposed alignment algorithm used in *ClustalW* consists of three main stages.

1. All pairs of sequences are aligned separately in order to calculate a distance matrix giving the divergence of each pair of sequences.

2. A guide tree is calculated from the distance matrix.

3. The sequences are progressively aligned according to the branching order in the guide tree.

Distance Matrix

The first step is to generate a set of pairwise distances. *ClustalW* offers two choices for aligning two sequences. The first one is a fast approximate method which allows to align a large number of sequences. The alignment scores are calculated as the number of k-tuple matches in the best alignment between two sequences minus a fixed penalty for every gap. The k-tuple consists of a run of identical residues, typically of length $1-2$ for proteins and of length $2-4$ for nucleotide sequences. The second choice is a slower but more accurate method. It uses two gap penalties (affine gap penalty): one for opening a gap and one for extending a gap. Since *ClustalW* is primarily intended for amino acid sequences, a full amino acid weight matrix such as the BLOSUM [Henikoff and Henikoff, 1992] and the PAM [Dayhoff et al., 1978] matrices can be used to score individual residues. However, other matrices can be used to score nucleotide sequences. The overall score of the alignment between two sequences is calculated as the number of identities in the best alignment divided by the number of residues compared (gaps are excluded). The resulting value is a similarity value that has to be transformed to a distance value by dividing the values by 100 and subtracting them from 1.0 to give the number of differences per site.

Guide Tree

The distances calculated in the distance matrix are used to guide the multiple alignment by means of the neighbor joining method [Saitou and Nei, 1987]. The neighbor joining method generates an unrooted tree. Its branch lengths correspond to estimated divergence of the sequences. The unrooted tree becomes a rooted tree by generating an artificially root node and placing it in the middle of a branch such that on either side the means of the branch lengths are equal. Sequences which have a common branch with other sequences share the weight of the same branch.

Progressive Alignment

Once the guide tree has been calculated, the multiple alignment is constructed by a series of pairwise alignments following the branching order in the guide tree. The closest two sequences, i.e. with the smallest distance value, are aligned first. This alignment is fixed and the next two sequences are aligned or a sequence is aligned with the first alignment, or, in general,

without sequence weights:

$$Score = (S(A,A) + S(A,C) + S(C,A) + S(C,C) + S(G,A) + S(G,C) + S(U,A) + S(U,C))/8$$

with sequence weights:

$$\begin{aligned} Score = (\quad & S(A,A) * w_1 * w_5 + S(A,C) * w_1 * w_6 + S(C,A) * w_2 * w_5 + S(C,C) * w_2 * w_6 + \\ & S(G,A) * w_3 * w_5 + S(G,C) * w_3 * w_6 + S(U,A) * w_4 * w_5 + S(U,C) * w_4 * w_6)/8 \end{aligned}$$

Figure 4.1: Two positions of the first alignment consisting of the sequences seq1, seq2, seq3 and seq4 and the second alignment consisting of the sequences seq5 and seq6 are compared. The score of two nucleotides is stored in matrix S. The scores are computed as the average of all pairwise nucleotide comparisons. The first score is a score without sequence weights, whereas the second score contains sequence weights denoted w_i.

two alignments are aligned. At each stage, a full dynamic programming is applied using a residue weight matrix and penalties for opening and extending gaps. The score at a position between a sequence or an alignment and a position between another sequence or another alignment is computed as the average of all pairwise weight matrix scores from the residues in the two sets of sequences. An example of two alignments is given in Figure 4.1.

Improvements

In the proposed *ClustalW* program from Thompson et al. [1994], there are a lot of improvements, which we only want to sketch. Details can be found in their article. Sequence weighting in the guide tree are performed such that the biggest weight gets a value of 1.0. Other weights are normalized relative to this value. *ClustalW* permits the usage of gap open penalties(GOP) and gap extension penalties(GEP). The gap penalties can be set according to the different weight matrices used and to the similarity of the sequences and to lengths including the length differences of the sequences. Gaps are

also dependent on the position specific residues. The GEP can be lowered in regions with many gaps. *ClustalW* is a program primarily intended for protein sequences, but nucleotide sequences can be also aligned due to their sequence similarities.

Example

Here, we demonstrate the alignment procedure of *ClustalW* by means of a set of 7 globins of known tertiary structure taken from Thompson et al. [1994]. The sequence names are from SwissProt [Bairoch and Apweiler, 2000]: Hba_Horse: horse α-globin, Hba_Human: human α-globin, Hbb_Horse: horse β-globin, Hbb_Human: human β-globin, Myg_Phyca: sperm whale myoglobin, Glb5_Petma: lamprey cyanohaemoglobin and Lgb1_Luplu: lupin leghaemoglobin. See Figure 4.2. Firstly, a 7×7 distance matrix between the seven globin sequences is calculated using the full dynamic programming algorithm. A neighbor-joining tree is constructed based on the distance values. This unrooted tree becomes a rooted tree by placing an artificial node in the middle of a branch such that on either side of this root the means of the branch lengths are equal. In Figure 4.2 the leghaemoglobin (Lgb2_Luplu) gets a weight of 0.442, which is equal to the length of the branch from the root to it. Another sequence such as the human *beta*-globin(Hbb_Human) gets a weight which is equal to the sum of all branch length from the root to it weighted with the number of shared branches. Hbb_Human gets the weight of

$$w = 0.081 + \frac{1}{2}0.226 + \frac{1}{4}0.061 + \frac{1}{5}0.015 + \frac{1}{6}0.062 = 0.221$$

The alignment proceeds according to the branching order of the guide tree, i.e. the rooted neighbor-joining tree. The progressive alignment is as follows: Hbb_Human vs. Hbb_Horse, Hba_Human vs. Hba_Horse, the two β-globins vs. the two α-globins, Myg_Phyca vs. the α- and β-globins, Glb5_Petma vs. the haemoglobins and the myoglobin, Lgb2_Luplu vas. the rest. The alignments are constructed under the agreements depicted in Figure 4.2.

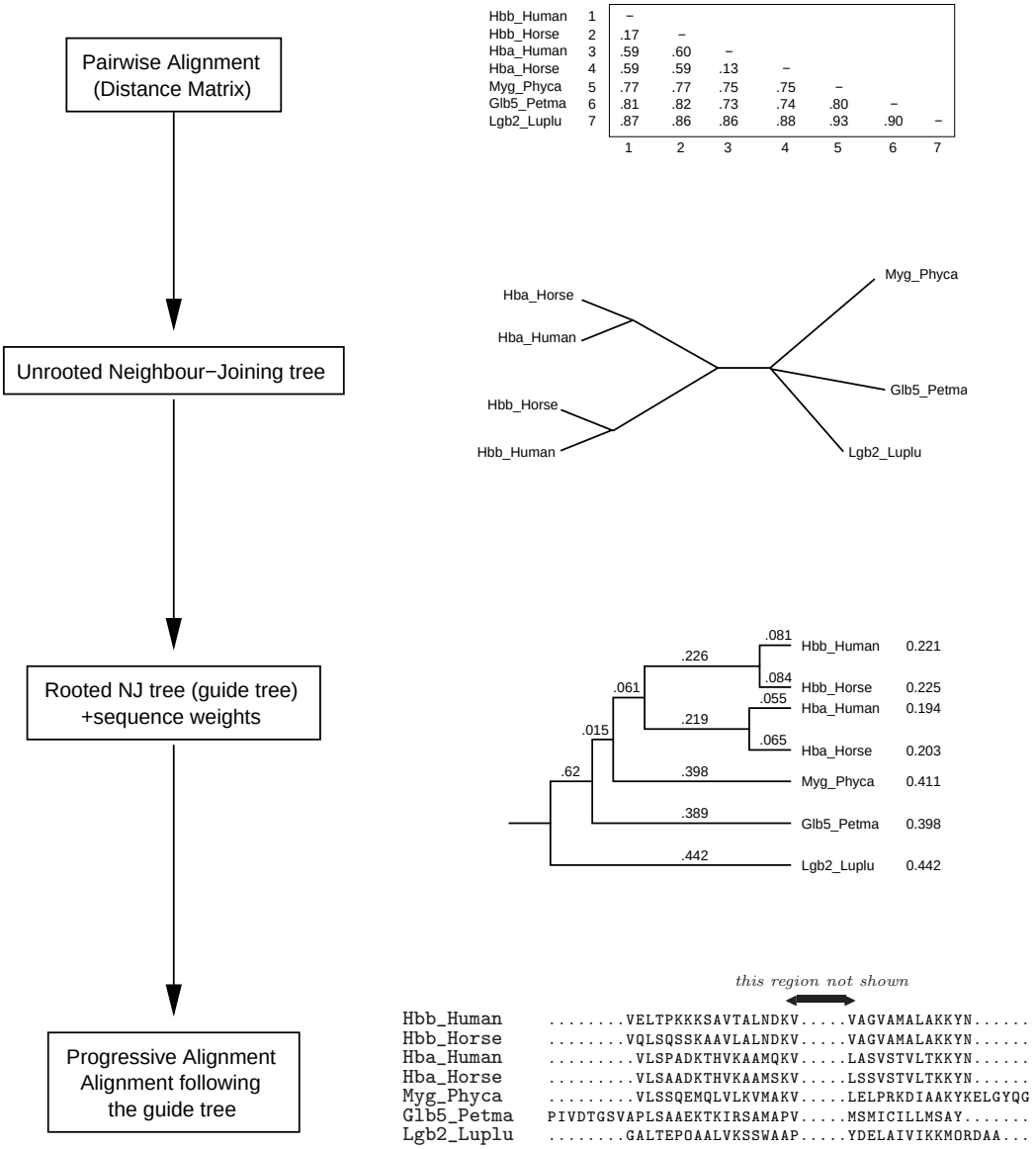


Figure 4.2: *ClustalW* alignment generated from the initial 7 globin sequences. The flow chart to the left hand side shows the *ClustalW* run performed on each set of sequences. Firstly, a distance matrix based on pairwise alignments is constructed, from which a neighbor joining tree is generated. The tree reflects the distances of the sequences dependent on the connections and on the weights of the branches. The rooted tree serves as the guide tree, that shows the order, in which the sequences are aligned. Lastly, the alignment is the result of the *ClustalW* program. Here, we see an excerption of the alignment.

T-Coffee

ClustalW is based on the progressive alignment strategy, a commonly used heuristic. Carefully choosing sensitive sequence weightings, position specific gap penalties and weight matrices results in high quality sequence alignments. Nevertheless, errors made in the first alignments cannot be rectified later as the rest of the sequences are added in following the guide tree. *T-Coffee* [Notredame et al., 2000] attempts to minimize that effect. *T-Coffee* (Tree-based Consistency Objective Function for Alignment Evaluation) can be characterized by mainly two features. The first one is to use heterogeneous data sources for generating multiple alignments. This means that a multiple alignment is constructed not only based on pairwise alignments as this is the case in *ClustalW*, but also on additional sources like e.g. local alignments. This information is collected in a so-called library from which a consistency based alignment can be constructed. The second feature of *T-Coffee* is to minimize alignment errors that typically occur in *ClustalW*. Pairwise comparisons are stored in the library with all these additional information (global alignments and/or local alignments from additional sources etc.). A processing step that transforms the library into an extended library contains all these information in pairwise alignments based on residue- and position- specific alignment weights.

Generating primary library of alignments

The library contains information of all pairwise sequences allowing several alignment sources. Notredame et al. [2000] propose two alignment sources for the pairwise comparison. The first one is a global alignment made by *ClustalW* [Thompson et al., 1994], and the second one calculates the ten top scoring non-intersecting local alignments using Lalign [Pearson and Lipman, 1988] with default parameters. In a set of N sequences the number of comparisons is $N(N-1)/2$. It is not required that the two alignment sources proposed by *T-Coffee* are consistent. Having computed all the alignment information the data is stored as a list of pairwise residue matches, i.e. a residue X in sequence i is aligned with residue Y in sequence j.

This residue alignment is weighted by using a weighting schema. The default weights of *T-Coffee* are derived by sequence similarities. This means that each individual residue pair gets a weight that is equal to the sequence identity in percent of the computed alignment. This holds for all two alignment sources, i.e. for *ClustalW* as well as Lalign.

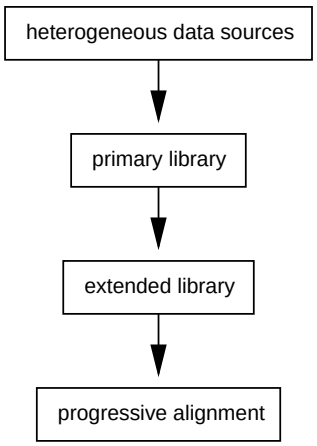


Figure 4.3: *T-Coffee* strategy given as a flowchart. The heterogeneous data sources can be any kind of comparable weighted sequence information. An example of global and local alignments is proposed by Notredame et al. [2000]. This information is collected in a so-called library which is processed and improved to the extended library.

Due to several alignment sources it is highly probable that residue pairs generated in the library occur more than once. In the library, each residue pair is represented once coming along with a weight that is the sum of the weights of this residue pair occurring in all alignment sources. If a residue pair does not occur then it receives a weight of zero, i.e. this is equivalent to not explicity listing in the library.

The list of weighted residue pairs is called the primary library ready to compute a multiple alignment. A dynamic programming algorithm allows us to find the best alignment. However, the alignment can be improved by examining the consistency of each residue pair with residue pairs coming from other alignments. The consistency of residue pairs can be weighted, and this process is called *library extension*.

Library extension

The library extension relies on a heuristic approach to assign a final weight on each residue pair that reflects some of the information contained in the whole library. Each residue pair is assigned a new weight consisting of the old weight plus a weight that reflects how the two residues align with the residues of the remaining sequences, i.e. other than the two sequences from which

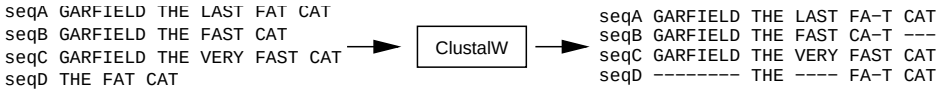
the two comparable residues stem. A triplet of residues consists of the two comparable residues and one additional residue from one of the remaining sequences. The aim is to determine the weights based on considering all triplets. If we have a look at Figure 4.4, then we see the four sequences A,B,C and D. Let $A(G)$ the G in $GARFIELD$ in sequence A and let $B(G)$ the G in $GARFIELD$ in sequence B. The initial weight of aligning the two $G's$ is 88 which is the sequence identity in per cent.

To determine the weight of a residue pair that takes into account other residues we see in Figure 4.4 that if we consider the alignment of the residue G in sequences A and B and if we also consider the G of sequence C then it makes sense to increase the initial weight of 88. The increment is determined by the minimum of the weight between $A(G)$ and $C(G)$ (weight=77) *and* $B(G)$ and $C(G)$ (weight=100). The final weight is thus $88 + 77 = 165$. The determination of all weights subject to the triplets is called library extension. The number of all triplets is N^3 in a set of N sequences. If we assume that all sequences have nearly the same sequence length L and if one pairwise comparison takes time $O(L^2)$, then the overall time complexity is $O(N^2L^2)$ for all pairwise comparisons. The number of all triplets is $O(N^3)$. In case of that all residues of a pair of sequences are inconsistent to the residues of the remaining sequences then the time complexity to generate all residue weights of the two sequences is closer to $O(L^2)$. In practice, this last step takes approximately linear time in L. The overall time complexity to build the library and to perform the library extension is $O(N^3L^2)$ in the worst case. However, for most cases where one wants to align homologue sequences, the precondition of consistency is given. Then, it takes time $O(N^2L^2)$.

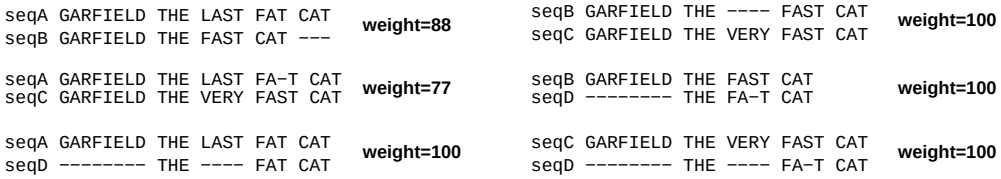
Progressive Alignment

The computation of the multiple alignment follows the idea of *ClustalW* from Thompson et al. [1994]. Pairwise distances of the sequence set are computed by using a dynamic programming approach to align the two sequences. Residue weights that are stored in the extended library are used for this task. The generated distance matrix is used to produce a neighbor-joining tree that guides the alignment process. The two closest sequences are aligned first. This alignment is fixed and the next closest sequence is aligned to this existing alignment or two new sequences are aligned or two existing alignments are aligned. In case of aligning an already existing alignment the average score in each column is taken. Gap penalties need not be set because

a) *ClustalW* Alignment



b)Pairwise Comparisons(primary library)



c)library extension

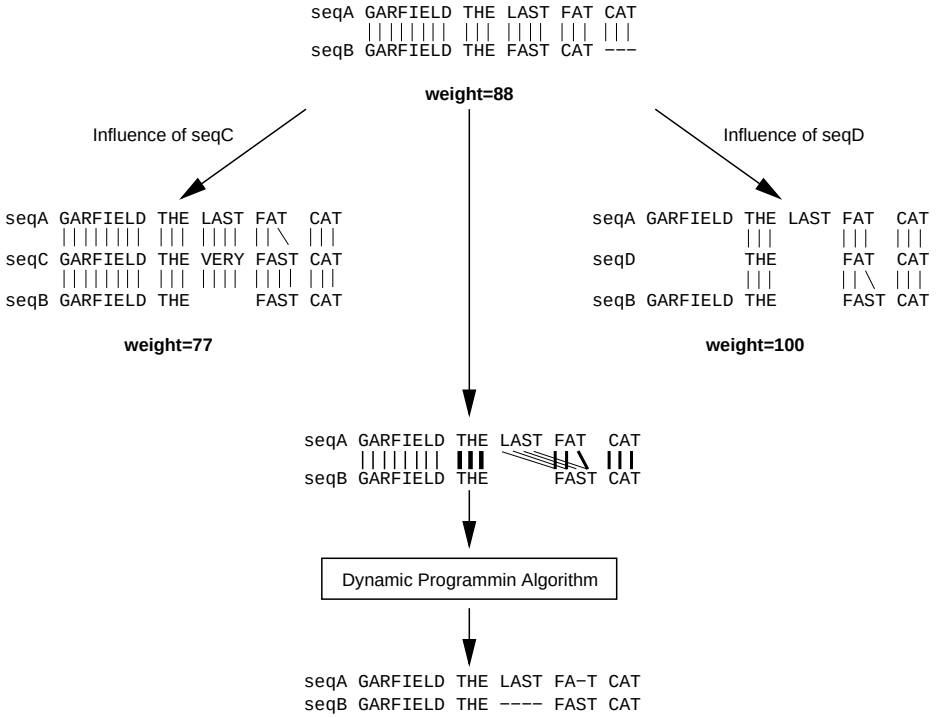


Figure 4.4: *T-Coffee* improvement vs. *ClustalW* alignment. a) Four sequences have been aligned using the standard *ClustalW* program with default values. Obviously, the word CAT in sequence seqB has been misaligned. b) Primary library: *T-Coffee* approach with pairwise comparisons made of *ClustalW* alignments. The weights are determined by the average identity among matched residues. c) Library extension: The first pairwise alignment of sequences seqA and seqB are considered to be how aligned with the remaining sequences, i.e. sequences seqC and seqD. The extended library contains all the additional information accompanied with their weighted alignment edges. A dynamic programming approach resolves the pairwise alignment of seqA and seqB. The word CAT is now aligned correctly.

they are already included in the alignment as sequence identity and residue weights, i.e. residues which are aligned with gaps get a weight of zero.

The complexity of the progressive alignment is composed of two steps: the first step is to construct the neighbor-joining tree which takes time $O(N^3)$ (see Saitou and Nei [1987]) and the second step is to align all these sequences which takes time $O(NL^2)$. Altogether, the time complexity can be formulated as

$$O(N^2L^2) + O(N^3L) + O(N^3) + O(NL^2)$$

4.2 Sequence Structure Alignments

4.2.1 Major Problems of Sequence Structure Alignments

Since the beginning of releasing multiple sequence alignment algorithms, the main focus was to align sequences without structural constraints. In the past years, a wave of multiple alignment techniques considering not only pure nucleotide sequences but also additional structural properties has been published. One of the most prominent and classical paper that introduces a dynamic programming algorithm to simultaneously align and fold RNA sequences stems from Sankoff [1985]. This paper addresses the main classical combinatorial RNA problem. Although this algorithm has a high computation time of $O(n^{3N})$ and memory usage of $O(n^{2N})$, where N is the number of sequences and n is the length of the longest sequence, the described problems serve as starting points for many heuristic approaches. One of the most popular and convincing idea has been published by Hofacker et al. [2004a]. They present a method to compute pairwise and progressive multiple alignments from the direct comparison of base-pairing probability matrices. Their proposed recurrence relation has the same time complexity of $O(n^6)$ and space complexity of $O(n^4)$ for a pairwise comparison. In contrast to the exponential solution given by Sankoff [1985] for multiple RNAs, the algorithm from Hofacker et al. [2004a] handles these RNA sequences in polynomial time. Despite its polynomial time it is only practical for short sequences. Another problem that arises here is the progressive alignment strategy. Usually, a distance tree is build that guides the alignment. First, two sequences are aligned. Then the next two closest sequences are aligned or a sequence is added to an already existing alignment or two existing alignments are

aligned. One of the commonest weaknesses in all progressive alignments is that these alignments rely on the fact 'once a gap, always a gap'. This means that if gaps are incorporated into intermediate alignments then they occur in the final alignment as well. In general, errors made in the first alignments cannot be rectified later as the rest of the sequences are added in.

We review the classical simultaneous alignment and folding approach by Sankoff [1985] and a faster approach based on base-pairing probability matrices by Hofacker et al. [2004a]. The idea of Wang and Zhang [2004] is an attempt to progressively align RNA structures by reducing the multiple structure alignment problem to the problem of aligning two RNA structures. Although not successful in practice, it clarifies the main drawbacks of such theoretical concepts.

An approach that minimizes the gap and error sensitivities caused by the usual progressive alignment strategy is to generate weighted alignment edges between pairwise nucleotide sequences and to progressively align them based on their alignment weights. It is implemented in the tool *MARNA* (Multiple Alignment of RNAs) available via the web page http://www.bioinf.uni-freiburg.de/Software/MARNA/index.html. This method became accepted in publicity due to their practical and elaborated theoretical concepts. *MARNA* is introduced in the section 4.3.

4.2.2 Simultaneous Aligning and Folding of Multiple RNAs

Sankoff [1985] proposes a fundamental dynamic programming algorithm to align and fold RNA sequences simultaneously. Basically, three single problems that consist of aligning, folding and reconstruction of ancestral sequences for N sequences are formulated in a mathematical framework. Each of them can be computed in polynomial time partially using recurrence equations. The combination of all problems is solved simultaneously for a set of N sequences of length n in time proportional to n^{3N} and storage proportional to n^{2N}. The three single problems are incorporated into one weighted objective function. Here, we briefly report the most important results. First, the simultaneous aligning and folding of a pair of RNA sequences is proposed. Finally, the addition of reconstructing ancestral sequences, i.e. the alignment and folding of N sequences is presented.

Pairwise Aligning and Folding

Alignment: Based on sequentially comparable nucleotides the alignment of two sequences S^1 and S^2 is composed of costs turning one sequence into the other using costs of insertion, deletion and replacement. The alignment is a dynamic programming solution. Its algorithm for computing all values $D(i, j; h, k)$ of the partial sequences $S_i^1, \ldots, S_j^1$ and $S_h^2, \ldots, S_k^2$ consists of applying recurrences which depend on the precalculated values $D(i, j; h, k-1)$, $D(i, j-1; h, k-1)$ and $D(i, j-1; h, k)$. The computation results in n^2m^2 entries. The sequence alignment problem can be solved by an $O(nm)$ solution of course. For the combination with the folding procedure, we will need all partial sequence values.

Folding: As already introduced in section 2.2, RNA secondary structures can be decomposed into loops named hairpins, bulges, interior loops, stacks and multi-loops. They are classified due to their accessible base-pairs. For each loop there exists an energy function. The thermodynamic parameters determines the stability of an RNA. However, for computing the minimum free energy structure of an RNA, there is an $O(n^3)$ algorithm (see also chapter 5).

Aligning and Folding: To solve the combination of both, a recursion equation is given that has its goal in aligning and folding two RNA sequences simultaneously. One aspect that has to be mentioned here is the *equivalence* of two structures. Two structures are said to be *equivalent* if both RNAs occupy two similar structures. This includes the requirement of the same branching order, i.e. if a multi-loop occurs in the first structure with k radiating stems, then this multi-loop with the same branching order is also found in the second RNA. This is a necessary constraint on branching loops. However, for bis-loops, there is no constraint against deleting or inserting their accessible pairs. In other words, the number of hairpins in both structures is the same.

The aim is to find two equivalent structures with sequential alignment constraints such that the combination of both single evaluation functions is expressed in one function and is optimal in some sense. The two single functions given as $D(i_1, j_1; i_2, j_2)$ and the energy functions are incorporated in the new objective function as the weighted sum.

The optimizing structure and alignment can be formulated in a theorem:

Theorem 5 *[Sankoff, 1985] Let $F(i_1, j_1; i_2, j_2)$ be the minimum cost possible for a pair of equivalent secondary structures S_1 and S_2 on positions $i_1, \ldots, j_1$*

and $i_2, \ldots, j_2$ of sequences $S^{(1)}$ and $S^{(2)}$, respectively, where the cost is the sum of the free energies and a constrained alignment cost.

Let $C(i_1, j_1; i_2, j_2)$ be the minimum cost given that $(i_1, j_1) \in S_1$ and $(i_2, j_2) \in S_2$ without considering the costs of aligning $S^{(1)}_{i_1}$, $S^{(1)}_{j_1}$, $a^{(2)}_{i_2}$ and $S^{(2)}_{j_2}$. If no such pair of structures exists, set $C = \infty$. Then

$C(i_1, j_1; i_2, j_2) =$

$$\min \begin{cases} e(s_1) + e(s_2) + D(i_1+1, j_1-1; i_2+1, j_2-1), \quad s_1, s_2 \text{ hairpins} \\ \qquad\qquad\qquad\qquad \text{closed by } (i_1, j_1), (i_2, j_2) \text{ respectively,} \\ \min\{e(s_1) + e(s_2) + C(p_1, q_1; p_2, q_2) \\ \qquad + D(i_1+1, p_1; i_2+1, p_2) + D(q_1, j_1-1; q_2, j_2-1)\}, \\ \qquad\qquad s_1, s_2 \text{ are bis-loops closed by } (i_1, j_1), (i_2, j_2) \\ \qquad\qquad \text{with } (p_1, q_1), (p_2, q_2) \text{ accessible,} \\ \qquad\qquad p_1 - i_1 + j_1 - q_1 - 2 \le U, \\ \qquad\qquad p_2 - i_2 + j_2 - q_2 - 2 \le U, \\ \qquad\qquad \text{or one of } \begin{cases} s_1 = \emptyset \quad \text{and} \quad (p_1, q_1) = (i_1, j_1) \\ s_2 = \emptyset \quad \text{and} \quad (p_2, q_2) = (i_2, j_2), \end{cases} \\ \min\limits_{\substack{i_1 < h_1 < j_1 - 1, \\ i_2 < h_2 < j_2 - 1}} \{G(i_1+1, h_1; i_2+1, h_2) \\ \qquad\qquad + G(h_1+1, j_1-1; h_2+1, j_2-1) + 2A\} \end{cases} \tag{4.3}$$

where
$G(i_1, j_1; i_2, j_2) =$

$$\min \begin{cases} C(i_1, j_1; i_2, j_2) + 2MB + D(i_1, i_1; i_2, i_2) + D(j_1, j_1; j_2, j_2), \\ \min\limits_{\substack{i_1 < h_1 < j_1, \\ i_2 < h_2 < j_2}} \begin{cases} G(i_1, h_1; i_2, h_2) + (j_1 - h_1 + j_2 - h_2)MC \\ \qquad + D(h_1+1, j_1; h_2+1, j_2), \\ G(i_1, h_1; i_2, h_2) + G(h_1+1, j_1; h_2+1, j_2), \\ (h_1 - i_1 + 1 + h_2 - i_2 + 1)MC \\ \qquad + G(h_1+1, j_1; h_2+1, j_2) + D(i_1, h_1; i_2, h_2) \end{cases} \end{cases} \tag{4.4}$$

and

$F(i_1, j_1; i_2, j_2) =$

$$\min \begin{cases} C(i_1, j_1; i_2, j_2) + D(i_1, i_1; i_2, i_2) + D(j_1, j_1; j_2, j_2), \\ \min\limits_{\substack{i_1 \le h_1 < j_1, \\ i_2 \le h_2 < j_2}} \{F(i_1, h_1; i_2, h_2) + F(h_1 + 1, j_1; h_2 + 1, j_2)\}, \\ D(i_1, j_1; i_2, j_2) \end{cases} \tag{4.5}$$

The recursion equation given in theorem 5 has time complexity $O(n^6)$ [Sankoff, 1985].

Multiple Aligning and Folding

In practice, RNA sequences come from N different organism. For that reason, an alignment of these sequences is proposed by Sankoff [1985]. It takes time proportional to n^{3N}. Suppose that the relationship of these RNA sequences is depicted as a tree such that terminal vertices represent the N different RNA sequences and the non-terminal vertices represent ancestral sequences showing the relationships of these given RNA sequences. Let $S^{a(1)}, \ldots, S^{a(N)}$ be the N RNA sequences and define

$$D(\overline{i, j}) = \min_{\text{not all } j'_r = j_r} \{D(\overline{i', j'}) + \gamma(\overline{\alpha_j(j - j')})\} \tag{4.6}$$

where $j'_r = j_r$ or $j'_r = j_r - 1$ for $r = 1, \ldots, N$ and $D(\overline{i, j})$ represents the alignment of the partial sequences $S_{i_1}^{a(1)}, \ldots, S_{j_1}^{a(1)}; \ldots; S_{i_N}^{a(N)}, \ldots, S_{j_N}^{a(N)}$. $\gamma(\overline{\alpha_j(j - j')})$ is an $N-$vector whose rth component is $S_{j_r}^{(r)}$ or $\emptyset$ depending on whether $j'_r = j_r - 1$ or $j'_r = j_r$. We omit here the relationships of these RNA sequences computed by a parsimony algorithm that can be depicted as a tree. Rather, we refer to Sankoff [1985].

Theorem 6 *[Sankoff, 1985] Let $F(\overline{i, j})$ be the minimum cost possible for N equivalent secondary structures $S_1, \ldots, S_N$ on positions $i_1, \ldots, j_1$; $i_2, \ldots, j_2$; $i_N, \ldots, j_N$ of sequences $S^{(1)}, S^{(2)}, \ldots, S^{(N)}$, respectively, this cost being the sum of the free energies and the alignment cost over a given phylogeny (tree). Let $C(\overline{i, j})$ be the minimum cost under the constraints $(i_1, j_1) \in S_1, \ldots,$ $(i_N, j_N) \in S_N$ but excluding the costs of aligning the elements of these closing pairs. If no such $N-$tuple of structure exists, then $C = \infty$. Then*

$$C(\overline{i,j}) =$$

$$\min \begin{cases} \sum e(s_r) + D(\overline{i+1,\ldots,j-1}), s_r \text{ the hairpin closed by } (i_r, j_r), \\ \min\limits_{\substack{\text{not all} \\ (p_r,q_r)=(i_r,j_r)}} e(s_r) + C(p,q) + D(\overline{i+1,p}) + D(\overline{q,j-1}), \\ \qquad s_r \text{ the bis-loop closed by } (i_r, j_r) \text{ with} \\ \qquad (p_r, q_r) \text{ accessible, } p_r - i_r + j_r + q_r - 2 \leq U, \\ \min\limits_{i_r<h_r<j_r-1} G(\overline{i+1,h}) + G(\overline{h+1,j-1}) + N(MA) \end{cases} \tag{4.7}$$

where

$$G(\overline{i,j}) = \min \begin{cases} C(i,j) + N(MB) + \gamma(S_{i_1}^{(1)}, \ldots, S_{i_N}^{(N)}) + \gamma(S_{j_1}^{(1)}, \ldots, S_{j_N}^{(N)}), \\ \min\limits_{i_r<h_r<j_r} \begin{cases} G(\overline{i,h}) + \sum(j_r - h_r)MC + D(\overline{h+1,j}), \\ G(\overline{i,h}) + G(\overline{h+1,j}), \\ \sum(h_r - i_r + 1)MC + G(\overline{h+1,j}) + D(\overline{i,h}) \end{cases} \end{cases} \tag{4.8}$$

and

$$F(\overline{i,j}) = \min \begin{cases} C(\overline{i,j}) + \gamma(S_{i_1}^{(1)}, \ldots, S_{i_N}^{(N)}) + \gamma(S_{j_1}^{(1)}, \ldots, S_{j_N}^{(N)}), \\ \min\limits_{i_r \leq h_r<j_r} \{F(\overline{i,h}) + F(\overline{h+1,j})\}, \\ D(\overline{i,j}) \end{cases} \tag{4.9}$$

with initial conditions $C(\overline{i,j}) = \infty$ *if any* $i_r = j_r$.

Recall the energy contribution of a multiple loop given as $e(ml) = MA + b * MB + c * MC$. Obviously, the time needed by this algorithm is proportional to n^{3N}. It is infeasible for practical data sets. Introducing a 'cutting corner' leads us to reduce the time complexity to $n^3(U^2W)^N$. Here, the idea of investigating the folding region in $i_1, \ldots, i_N$ is restricted to the idea of investigating the i_r for each $i = 1, \ldots, n$ with

$$i - \frac{W}{2} \leq i_r \leq i + \frac{W}{2}$$

Similarly, this is done for h and j.

4.2.3 Faster Approach by Aligning Base-Pairing Probability Matrices

The idea of folding and aligning RNA sequences is supposed to be one of the most challenging tasks in RNA structure analysis. As we have seen in the last chapter, the mathematical framework given by Sankoff [1985] is infeasible with current computational resources. Instead of attempting to solve the folding and alignment problem simultaneously by means of energetic parameters, Hofacker et al. [2004a] uses the approach of McCaskill [1990] to first compute base-pairing probability matrices and then to apply a simplified variant of Sankoff's algorithm. Base-pairing probability matrices include information about RNA energetics. The pairwise folding and alignment algorithm as proposed by Sankoff [1985] is an $O(n^6)$ time and $O(n^4)$ memory algorithm. The original problem description given by Hofacker et al. [2004a] has indeed the same complexity, but is capable of reducing the time complexity to $O(n^5)$ by restricting the span of matching pairs to a limited size. The intrinsic improvement is in aligning multiple RNA sequences.

Scoring Function for Pairwise RNA Alignments

Instead of considering two RNA sequences in conjunction of the thermodynamic loop-based energy model, the base-pairing probability matrices as proposed by McCaskill [1990] are precomputed and the sequences are aligned due to their matrices. The precomputation needs time $O(n^3)$ for each RNA. Suppose that two RNA sequences A and B are given with their probability matrices P^A and P^B. The aim is then to find two similar structures occurring with high probabilities. This can be formulated as to find a list of base-pair matches $(i, j) \in A$ and $(k, l) \in B$ such that they form a secondary structure:

$$\sum_{matches(i,j;k,l)} (\psi_{ij}^A + \psi_{kl}^B) \rightarrow \max \tag{4.10}$$

Here, ψ_{ij}^A is the weight of the base-pair (i, j) in sequence A. It can be set to any value favorizing the base-pair probabilities. To determine the significant value its value is finally set to $\psi_{ij}^A = log(\frac{P_{ij}}{p_{min}})$ with p_{min} as the significant value. ψ_{ij}^A is calculated using the log-odds scoring scheme with the baise-pairing probability as the 1-model and p_{min} as the 0-model.

If we also consider a gap penalty $\gamma < 0$ (preferably a value between -3 and -5 as Hofacker et al. [2004a] suggest) and the value N_{gap} referring to

the number of insertions and deletions between these two sequences then the alignment of the sequences can be formulated as a mathematical formula:

$$\sum_{(i,j;k,l)\in S} [\psi_{ij}^{A} + \psi_{kl}^{B} + \tau(A_i, A_j; B_k, B_l)] + \gamma N_{gap} + \sum_{i\in A, k\in B \notin S} \sigma(A_i, B_k) \rightarrow \max \tag{4.11}$$

$\tau(A_i, A_j; B_k, B_l)$ is the substitution score of transforming the base-pair $(i, j) \in A$ into the base-pair $(k, l) \in B$. $\sigma(A_i, B_k)$ is the substitution score of transforming the unpaired base A_i into the unpaired base B_k. For the simplest case, both functions can be set to 0 if the sequential properties should be disregarded, i.e. the function is left to construct a structural alignment. An alternative is to use the parameters from the *RIBOSUM* matrix [Klein and Eddy, 2003], or from any other sources containing substitution rates.

Algorithm to align two RNAs

With this proposed scoring scheme, it is easy to derive a recurrence relation for aligning and folding two RNAs based on the base-pairing probability matrices. So let $S_{i,j;k,l}$ be the score of the best match of the subsequences $A[i..j]$ and $B[k..l]$. Let $S^{M}_{i,j;k,l}$ be the best match subject to the constraint that (i, j) and (k, l) are matched base-pairs. Then the recurrence relation is given as:

$$S_{i,j;k,l} = \max \begin{cases} S_{i+1,j;k,l} + \gamma \\ S_{i,j;k+1,l} + \gamma \\ S_{i+1,j;k+1,l} + \sigma(A_i, B_k) \\ \max\limits_{h\le j, q\le l} (S^{M}_{i,h;k,q} + S_{h+1,j;q+1,l}) \end{cases} \tag{4.12}$$

$$S^{M}_{i,j;k,l} = S_{i+1,j+1;k+1,l+1} + \psi_{ij}^{A} + \psi_{kl}^{B} + \tau(A_i, A_j; B_k, B_l) \tag{4.13}$$

$S_{i,j;k,l}$ is initialized as follows: $S_{i,j;k,l} = |(j-i)-(l-k)|\gamma$ for $j-i \le M+1$ or $l-k \le M+1$. M is the minimum size of a hairpin loop (mostly set to 3). Obviously, the equation 4.12 determines the alignment by deleting a base (first line), inserting a base (second line), substituting a base (third line) or matching a base-pair (fourth line). It is clear that this algorithm needs a

memory capacity proportional to n^4 because of its four variables i, j, k and l in $S_{i,j;k,l}$ and a time complexity of $O(n^6)$ because of computing each cell in $S_{i,j;k,l}$ which takes n^4 considerations, each of them has to be computed by the fourth line that consists of running two additional variables h and q depending on each other and both restricted to n, the length of the RNAs.

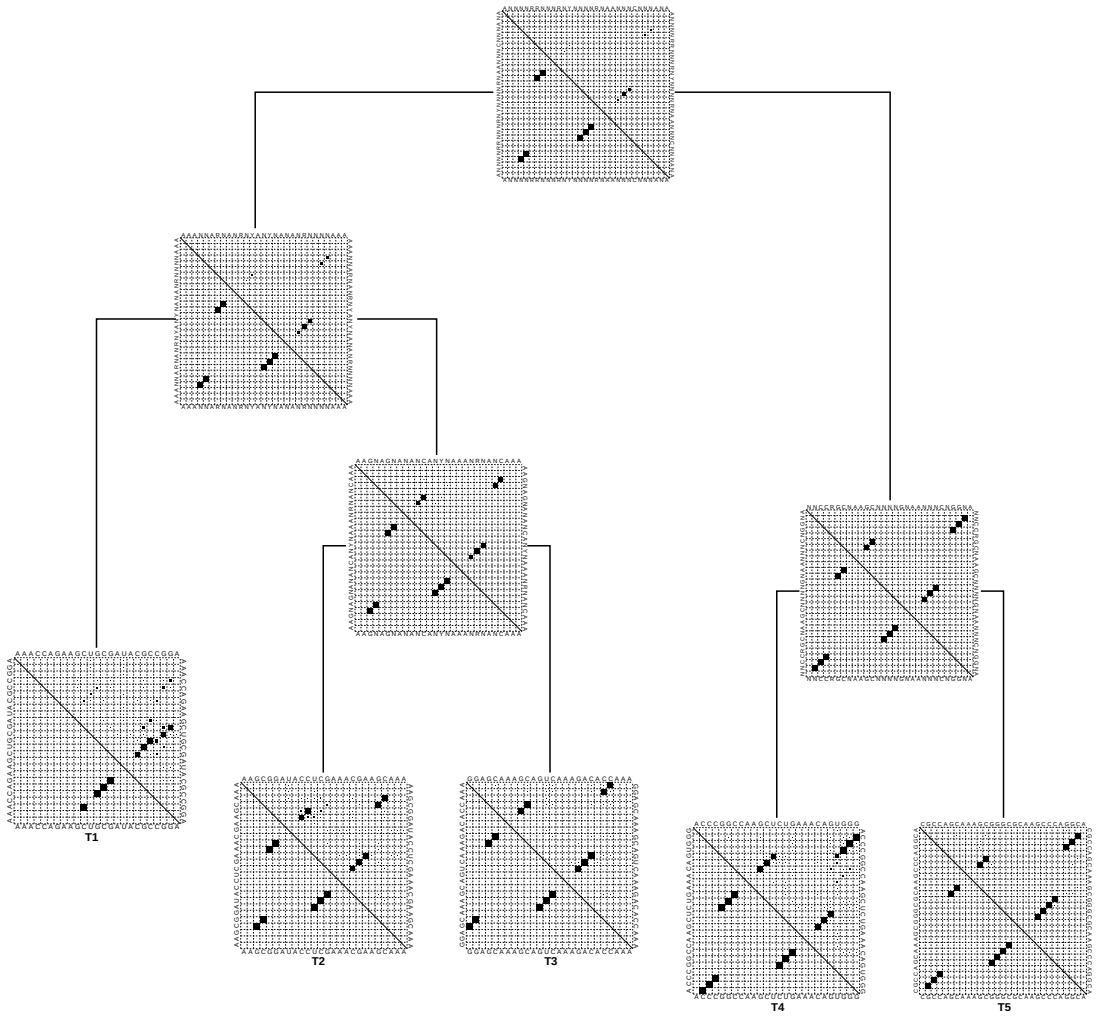
Here, the algorithm can be improved to an $O(n^5)$ algorithm by restricting the size of the maximal span match of the base-pairs $(i, j) \in A$ and $(k, l) \in B$ written as $\Delta = |(j-i)-(l-k)|$. Retrieving the alignment results in a standard backtracking algorithm consisting of matching unpaired bases, inserting or deleting gaps and matching base-pairs with other base-pairs or with gaps.

Aligning Multiple RNAs

The alignment of two RNAs is implemented in a program called *PMcomp*. The quantified values of *PMcomp* are similarity values. It is used to construct a guide tree to align multiple RNAs with the weighted pairgroup clustering method. The implementation that performs the multiple alignment is called *PMmulti*. For the multiple step, it is also necessary to align either two sequences, or one sequence with an alignment or two alignments. Therefore, to evaluate two alignments, the probability matrices are computed by using mean probability matrices of two sequences and/or two alignments. Suppose the matrices are given as P^A and P^B. Then the 'consensus' or 'average' base pair-probability matrix is given as

$$P^{AB}_{p,q} = \begin{cases} \sqrt{P^A_{i_p,j_q} P^B_{k_p,l_q}} & \text{for matches,} \\ 0 & \text{for gaps} \end{cases} \tag{4.14}$$

where i_p is the position of sequence A corresponding to the position p of the alignment. If at least two sequences have been aligned, we can therefore deal with the mean base-pair probability matrices. Here, the geometric mean has been used, but other mean values, especially the arithmetic mean $(P^A_{i_p,j_q} + P^B_{k_p,l_q})/2$, are useful as well. Of course, the construction of the guide tree consists of the comparison the N sequences resulting in $N(N-1)/2$ similarity values. In practice, the execution of *PMmulti* is restricted to short sequences. An example of such an alignment is depicted in Figure 4.5




```
T1              AAACC-AGAAGCU-GCGAU-ACGC-CGG--A
                ...((-((...))-(((..-.)))-.))--.
T2              -AAGC-GGAUACC-UCGAA-ACGA-AGCAAA
                -..((-((...))-(((..-.)))-.))...
T3              ---GGAGCAAAGCAGUCAA-AGAC-ACCAAA
                ---((.((...)).(((..-.)))-.))...
T4              -ACCCGGCCAAGCUCUG-AAACAG-UGG-G-
                -.(((.((...)).(((-...)))-.))-)-
T5              -CGCCAGCAAAGCGGGCGCAAGCCCAGG-CA
                -.(((.((...)).(((....)))..))-).
CONSENSUS_SEQ   ANNNNNNNNNNNNNNNNNNAANNNCNNNANA
CONSENSUS_STR   ~..((.((...)).(((..~.)))~.))~..
```

Figure 4.5: Toy example from Hofacker et al. [2004a] adapted and slightly modified. 5 sequences are aligned due to their base-pairing matrices. The lower left part of the matrices depicts the minimum free energy. The tree is computed with *PMmulti* which calls *PMcomp* (pairwise comparison) with a gap penalty of $\gamma = -0.5$. The alignment as well as the structures are shown below. There is a clear consensus sequence and consensus structure observable.

4.2.4 Progressive Multiple RNA Structure Alignment

Recently, Wang and Zhang [2004] published two heuristic algorithms to align multiple RNA structures based on the computational model as given by Wang and Zhang [2001]. The problem of aligning multiple RNAs is reduced to the problem of aligning two RNA structure alignments. It is rather a structure alignment problem than a sequence structure problem. This means that RNAs are primarily aligned due to their structures with fine-tuned sequence considerations rather than sequence *and* structure considerations. Nevertheless, Wang and Zhang [2004] apply techniques known from multiple sequence alignments. Based on the previous work by Ma et al. [2003],Gotoh [1982] and Kececioglu and Zhang [1998], the proposed algorithms solve the problem of progressive alignment by "aligning two structure alignments" similar to solving the problem of "aligning sequence alignments".

Edit Operations on RNA Structures

The edit operations on RNA secondary structures as given by Wang and Zhang [2004] resemble the ones given by Bafna et al. [1995] (see section 3.1.2). A base-pair is handled as an entity. We have operations on bases and on base-pairs. Edit operations on bases consist of deletion, insertion and substitution. The minimum cost of transforming one RNA into the other is the minimum edit distance. Edit operations on base-pairs are deletion (deletion of two bases), insertion (insertion of two bases) and substitution (substitution of two single bases). An example of a pairwise structure alignment with their corresponding edit operations is given in Figure 4.6

A corresponding problem known from sequence alignment techniques is the weighting of alignments dominated by their included gaps mostly denoted '-'. Affine gap penalty is to assign a cost of l consecutively followed gaps. To describe the cost, the affine function $a + b \times l$ is used where a is the initiation penalty, b is the extension penalty and l is the gap length.

Constraints on Structure Alignment

Let A be an alignment of multiple RNA structures and let $A[i]$ be the ith column, let $A[i, j]$ be the columns $A[i]$ and $A[j]$ and let $A_k[i]$ denote the ith character of the kth structure.

Once a multiple alignment is given, it can be scored using the most frequent function to score such alignments, namely the sum-of-pair score which is defined as

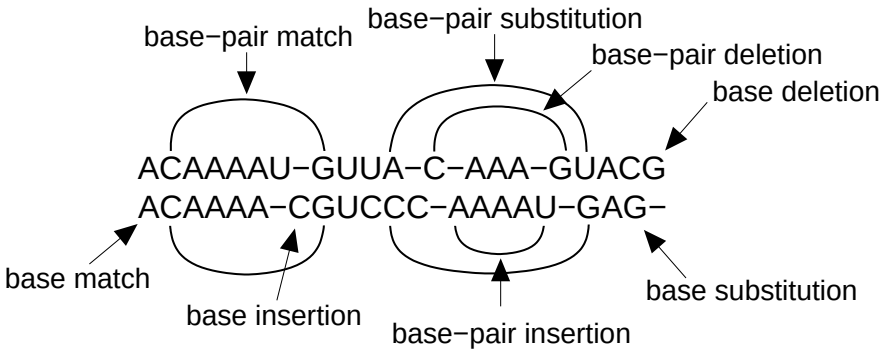


Figure 4.6: Edit operations consisting of operations on bases and on base-pairs used for RNA structure alignment.

$$score(A) = \sum_{i,j=1}^{n} ind\_pair(A_i, A_j) \tag{4.15}$$

where $ind\_pair(A_i, A_j)$ is the pairwise score between the ith RNA and the jth RNA and the alignment A consists of the aligned A_i's for $i = 1, \ldots, n$.

For the multiple alignment of RNA structures, we are interested in alignments of two structure alignments. In other words, the multiple structure alignment problem is reduced to the problem of aligning two RNA structure alignments. In this definition, columns are distinguished between single columns, base-pairing columns and gap columns:

Definition 5 *Suppose that we are given two alignments of RNA structures A and B, the alignment of them is represented by (A', B'), which should satisfy the following constraints:*

1. *A' is A with some gap columns inserted, B' is B with some gap columns inserted, and $|A'| = |B'|$.*

2. *If $A'[i]$ is a single column of A', then $B'[i]$ of B' is either a single column or a gap column; if $B'[j]$ is a single column of B', then $A'[j]$ of A' is either a single column or a gap column.*

3. *If $A'[i]$ and $A'[j]$ are pairing columns of A', then $(B'[i], B'[j])$ of B' are either pairing columns or two gap columns; if $B'[i]$ and $B'[j]$ are pairing*

columns of B', then $(A'[i], A'[j])$ of A' are either pairing columns or two gap columns.

Recall that the score of one RNA structure alignment A is given as $score(A) = \sum_{i,j=1}^{n} ind\_pair(A_i, A_j)$. The score of two alignments A with K structures and B with L structures is given as the quality Q defined as

$$Q(A', B') = \sum_{i=1}^{K} \sum_{j=1}^{L} ind\_pair(A'_i, B'_j)) \tag{4.16}$$

Aligning Structure Alignments

To formulate the structure alignment recursion, we first have a look at how to handle the gap penalties. There are two cases to consider as shown in the following example:

```
seq1:     x----         x----
seq2:     x---x         xx--x
              *             *
```

The marked column ('*') in the first alignment determines an alignment where we have a gap open penalty. The marked column ('*') in the second alignment determines an alignment where we can assign a gap extension penalty considering an induced alignment. There, we are faced with two different situations. The first one is given as a *pessimistic*, the second one is given as an *optimistic* gap opening.

Here, we consider two structure alignments given by A and B. The computation of the edit operations can be described by recursion equations:

- $D[i_1, i; j_1, j]$ is the alignment score between $A[i_1, i]$ and $B[j_1, j]$ ending with a deletion at $A[i]$.
- $I[i_1, i; j_1, j]$ is the alignment score between $A[i_1, i]$ and $B[j_1, j]$ ending with an insertion at $B[j]$.
- $M[i_1, i; j_1, j]$ is the alignment score between $A[i_1, i]$ and $B[j_1, j]$ ending with a substitution of $A[i]$ with $B[j]$.

Our objective is to find an optimal alignment based on the three edit operations insertion, deletion and substitution on structural alignments with minimal costs.

Suppose we have two structure alignments A and B with lengths R and S, respectively, and let P and Q the number of pairs of pairing columns, then we can state the algorithm to align both alignments:

```
STRUCTURE ALIGNMENT(A, B)
 1  Sort the pairing columns of A and B by 3'end in increasing
 2  order, respectively
 3  for each pair of pairing columns (pl,pr) in A
 4     do for each pair of pairing columns (ql,qr) in B
 5           do Compute the alignment of A[pl+1,pr-1] and B[ql+1,qr-1]
 6              by using lemmas and equations, and store the value in
 7              an array
 8  Compute A[1,R] and B[1,S] and
 9  output min{M(1,R;1,S),D(1,R;1,S),I(1,R;1,S)}
10  Trace back arrays M,I and D to find the alignment of A and B
```

All lemmas and equations are listed in Wang and Zhang [2004]. We state the concluding theorem:

Theorem 7 *The alignment of two RNA secondary structures A and B as proposed by Wang and Zhang [2004] could be computed optimally under the definitions of pessimistic and optimistic assumptions in time $O(R{\times}S{\times}P{\times}Q)$ and space $O(R \times S + R \times K + S \times L)$.*

4.3 MARNA: A Server for Multiple Alignment of RNAs

4.3.1 Classification

Several pairwise sequence structure alignment methods have been developed as described in the last chapters. They use extended alignment scores that evaluate secondary structure information in addition to sequence information. However, two problems for the multiple alignment step remain. First, how to combine pairwise sequence structure alignments into a multiple alignment and second, how to generate secondary structure information for sequences whose explicit structural information is missing.

Here, we describe a novel approach for multiple alignment of RNAs (*MARNA*) taking into consideration both the primary sequences and the secondary structures. It is based on pairwise sequence structure comparisons of RNAs as proposed by Jiang et al. [2002] (see section 3.1.3). From these sequence structure alignments, libraries of weighted alignment edges are generated. The weights reflect the sequential and structural conservation. For sequences whose secondary structures are missing, the libraries are generated by sampling low energy conformations. The libraries are then processed by the *T-Coffee* system [Notredame et al., 2000]. Furthermore, *MARNA* is able to extract a consensus-sequence and -structure from a multiple alignment.

Multiple sequence- and structure-based alignments of RNAs can be divided into two major classes, the probabilistic and the non-probabilistic approaches. Probabilistic approaches are based on stochastic context-free grammars (SCFG) and require an initial multiple alignment as input. The quality of the outputs crucially depends on this initial alignment. They are used to model RNA-families and/or to predict a secondary structure via comparative analysis, e.g. *Cove* [Eddy and Durbin, 1994], *RNACAD* [Brown, 1999b] and *Pfold* [Knudsen and Hein, 2003]. This kind of multiple alignments has its own topic in bioinformatics and is not considered in this thesis. A non-probabilistic, comparative approach is e.g. given by RNAlign [Corpet and Michot, 1994] that performs an alignment between a bank of aligned sequences and a new sequence.

Here, a non-probabilistic approach is proposed to align a set of more than two RNAs with or without known conformations. The standard approach is to perform direct pairwise alignments of RNAs using sequence and (sec-

ondary) structure information and to combine the pairwise alignments into a multiple alignment. No general approach yet exists even though there is a wealth of approaches for pairwise alignments of RNAs. The reason is that the results of the pairwise sequence structure alignments cannot simply be aligned in a progressive way (like e.g. profiles for sequence alignments). There is only one exception, namely *PMcomp/PMmulti* [Hofacker et al., 2004a] (see section 4.2.3).

4.3.2 Pairwise Alignment Scores

An alignment A of two sequence-structures $S_1 = (S_1, P_1)$ and $S_2 = (S_2, P_2)$ is a subset of $[1 \dots |S_1|] \cup \{-\} \times [1 \dots |S_2|] \cup \{-\}$, where for all pairs $(i, j), (i', j') \in A$ holds

1. $i \leq i' \Rightarrow j \leq j'$
2. $i = i' \neq - \Rightarrow j = j'$, and
3. $j = j' \neq - \Rightarrow i = i'$.

In addition, we require that for every $i \in [1 \dots |S_1|]$ there is some j with $(i, j) \in A$, and vice versa for $j \in [1 \dots |S_2|]$. The pairs $(i, j) \in A$ are called *alignment edges*. We say that $i \in [1 \dots |S_1|]$ is *aligned* with j if $(i, j) \in A$, and analogously for $j \in [1 \dots |S_2|]$. An alignment edge $(i, j) \in A$ is called *realized* if neither $i = -$ nor $j = -$.

The scoring of an alignment A of two sequence-structures $S_1 = (S_1, P_1)$ and $S_2 = (S_2, P_2)$ is based on the notion of edit operations on bases *as well as* on arcs. We present a slightly modified version of the distance-based scoring scheme from Jiang et al. [2002] (see also section 3.1.3):

Edit operations on free bases are *base match*, *base mismatch* and *base deletion*. A base match has cost 0, a base mismatch and a base deletion have positive costs. We combine these cost functions into a single cost function $w_{base}(i, j)$, where $w_{base}(i, j) = 0$ if and only if $S_1[i] = S_2[j]$. We will feel free to write either the positions or the nucleotides as arguments in the cost function where necessary.

For arcs, we have a more complex scoring scheme. Consider an arc $(i, i') \in P_1$ such that i is aligned with j and i' is aligned with j'. An *arc match* occurs if j, j' form an arc $(j, j') \in P_2$, $S_1[i] = S_2[j]$, and $S_1[i'] = S_2[j']$. We have an *arc mismatch* if $(j, j') \in P_2$, but $S_1[i] \neq S_2[j]$ or $S_1[i'] \neq S_2[j']$. Arc matches

have cost 0, whereas arc mismatches have cost $w_{am}(i, i', j, j')$. On the other hand, if $(j, j') \notin P_2$, then we have an *arc deletion* with cost $w_{ad}(i, i', j, j')$. Jiang et al. [2002] subdivided arc deletions into arc breakings, arc alterings and arc removings. An *arc breaking* occurs if none of j and j' equals the gap symbol $-$. If exactly one of j, j' equals $-$, then we have an *arc altering*. If both j, j' are equal to $-$, then we have an *arc removing*. The edit operations on arcs are summarized in Figure 3.3.

The total score of an alignment is the sum of costs of all applied edit operations that transform one sequence-structure into the other. The complexity of finding an alignment with minimal costs is determined by the way arc deletions are scored. Jiang et al. [2002] presented a dynamic programming algorithm that solves this problem in $O(n^2m^2)$ time and $O(nm)$ space under certain restrictions on the scoring of arc deletions. In fact, this requires the existence of functions $w^l_{ad}(i, j)$ and $w^r_{ad}(i', j')$ for the left and right ends, respectively, such that

$$w_{ad}(i, i', j, j') = w^l_{ad}(i, j) + w^r_{ad}(i', j').$$

In the following, we will not distinguish between left and right ends of an arc, i.e. we set $\forall i, j : w^l_{ad}(i, j) = w^r_{ad}(i, j) = w^e_{ad}(i, j)$, where $w^e_{ad}(i, j)$ is a single function to score both ends of an arc.

The effect of this restriction is that one can evaluate both arc ends in an alignment independently, which is a necessary prerequisite for the dynamic programming algorithm. This situation is depicted in Figure 4.7.

In our approach, we even simplify the scoring scheme further by defining $w^e_{ad}(i, j)$ to be composed of a base match, base mismatch or base deletion together with a fixed cost for deleting an arc. Hence, we set

$$w^e_{ad}(i, j) = w_{base}(i, j) + \frac{1}{2} w^{const}_{ad},$$

where w^{const}_{ad} is the cost for deleting one arc.

4.3.3 Multiple Alignment

Once the sequence structure alignments have been calculated for all pairs of input sequences, we construct the so-called *library*. A library for a pairwise alignment of two sequence-structures consists of the set of all realized edges together with a weighting of each edge. Then, the libraries for all pairwise

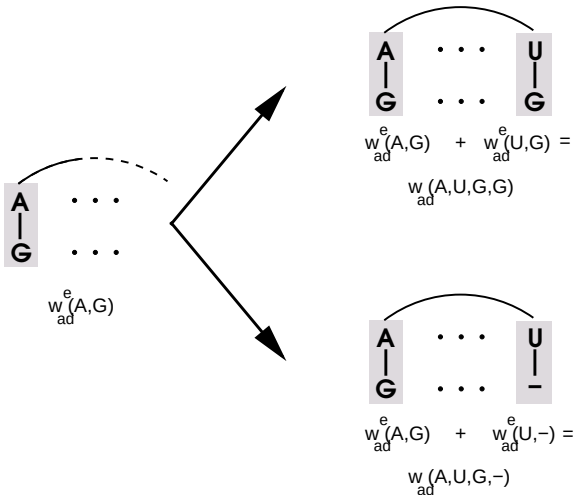


Figure 4.7: Independent scoring of both arc ends connecting base-paired nucleotides.

alignments are given to *T-Coffee* [Notredame et al., 2000] to build a single multiple alignment (see also section 4.1).

The *T-Coffee* system is a consistency based alignment method that combines local and global information to produce a multiple alignment in the following manner (see also section 4.1). First, an extended primary library is produced that improves all pairwise alignments by taking into consideration how all other sequences align with the current two RNAs. Edges achieve higher weights if the bases at the end points of these edges are also aligned with other sequences. Second, the improved data set of pairwise alignments is processed by a progressive alignment strategy. A distance matrix is computed between all sequences using the improved weights of alignment edges. Subsequently, the neighbor-joining method [Saitou and Nei, 1987] provides a phylogenetic tree, which dictates the order of aligning these sequences. Since the initial libraries were generated from sequence structure alignments, the resulting multiple alignment reflects the sequential *and* structural similarities of RNAs.

Distance and Similarity

The weights attached to realized edges in the libraries correspond to similarity weights. For that reason, we have to transform the distances defined

in the previous section into similarity values. Smith and Waterman [1981a] solved the problem of transforming distances into similarities for edit operations on bases. We extend this approach to our set of edit operations. The main observation of Smith and Waterman [1981a] is that one has to consider the number of nucleotides r involved in an edit operation. We call this number the *order* of the edit operation. In our case, we have edit operations with $r = 4$ (arc match and arc mismatch), $r = 2$ (base match and base mismatch) and $r = 1$ (base deletion). Since we have split the arc deletion operation into two separate edit operations for the arc ends, we have an edit operation with $r = 2$, if the arc end is aligned with a nucleotide, and an edit operation with $r = 1$ if the arc end is aligned with $-$.

By enumerating all different edit operations, we can write the distance score of an alignment A as

$$\text{dist}(A) = \sum_r \sum_k w^{r,k} \lambda_A^{r,k},$$

where $w^{r,k}$ is the cost for the kth edit operation of order r (for $r = 4, 2, 1$), and $\lambda_A^{r,k}$ is the number of times the kth edit operation of order r is used in the alignment A. Then we can rewrite the distances $w^{r,k}$ into similarities $s^{r,k}$ as follows:

Theorem 8 *Consider a scoring scheme where $w^{r,k}$ is the cost of the kth edit operation of order r. Let A^{MSP} be any fixed value, which is interpreted as the maximal similarity per nucleotide position we want to achieve. Define the similarity $s^{r,k}$ for the kth edit operation of order r by*

$$s^{r,k} = r \cdot A^{MSP} - w^{r,k}.$$

Then the alignment A which minimizes dist(A) is the alignment that maximizes sim$(A) = \sum_r \sum_k s^{r,k} \lambda_A^{r,k}$, and vice versa.

Proof 2 *The optimal alignment for two sequence-structures $\mathcal{S}_1 = (S_1, P_1)$ and $\mathcal{S}_2 = (S_2, P_2)$ under the similarity score is given by*

$$
\begin{aligned}
A_{opt} &= \underset{A \ align. \ of \ \mathcal{S}_1, \mathcal{S}_2}{\arg\max} \{\sum_{r,k} s^{r,k} \lambda_A^{r,k}\} \\
&= \arg\max_A \{\sum_{r,k} (r A^{MSP} - w^{r,k}) \lambda_A^{r,k}\} \\
&= \arg\max_A \{A^{MSP} \sum_{r,k} r \lambda_A^{r,k} - \sum_{r,k} w^{r,k} \lambda_A^{r,k}\} \qquad (4.17)
\end{aligned}
$$

Since any nucleotide position is involved in exactly one edit operation, we know that $\sum_{r,k} r\lambda_A^{r,k}$ *is the total number of nucleotide positions involved in edit operations. Hence,* $\sum_{r,k} r\lambda_A^{r,k} = |S_1| + |S_2|$*. Thus,*

$$
\begin{aligned}
A_{opt} &= \arg\max_A \{A^{MSP}(|S_1| + |S_2|) - \sum_{r,k} w^{r,k} \lambda_A^{r,k}\} \\
&= \arg\max_A \{-\sum_{r,k} w^{r,k} \lambda_A^{r,k}\} \ = \ arg\ min\{\sum_{r,k} w^{r,k} \lambda_A^{r,k}\} \qquad (4.18)
\end{aligned}
$$

Thus, one has only to choose the maximal similarity per position A^{MSP} to transform the distance score into a similarity score without changing the *global* optimal alignment. Although it does not change the global optimal alignment, it is important for the *T-Coffee* system since only the realized edges are considered when combining the pairwise alignments into a multiple alignment. This implies that alignment edges containing a gap have a weight of 0. To achieve a good approximation to this, we set

$$A^{MSP} = \max_{r,k} \{\frac{w^{r,k}}{r}\}.$$

The above theorem can also be extended to vary the contribution from structural to sequential positions for the generation of the multiple alignment. Obviously, the distance score is flexible enough to strengthen either structural or sequential positions. Structural positions are strengthened by rising the constant cost for arc deletion (i.e. w_{ad}^{const}). But this is somewhat lost if we have the same maximal similarity for structural and sequential positions. This leads to the following modification of the theorem. We say that a position i in the sequence-structure $\mathcal{S} = (S, P)$ is a structural position if there is an i' with $(i, i') \in P$ or $(i', i) \in P$. The position i is defined to

be *sequential* otherwise. The order of an edit operation is now defined by two values r_{str} and r_{seq}, which are the numbers of structural and sequential positions in the edit operation, respectively. For an alignment A the value $\lambda_A^{r_{str},r_{seq},k}$ denotes again the number of times the kth edit operation of order r_{str}, r_{seq} is used in A. Then we can write the distance score of A as $\text{dist}(A) = \sum_{r_{str},r_{seq},k} w^{r_{str},r_{seq},k} \lambda_A^{r_{str},r_{seq},k}$.

Theorem 9 *Let $w^{r_{str},r_{seq},k}$ be the cost of the kth edit operation of order r_{str}, r_{seq}. Let A_{str}^{MSP} be the maximal similarity for structural positions, and let A_{seq}^{MSP} be analogously defined for sequential positions. Define the similarity for the kth edit operation of order r_{str}, r_{seq} by*

$$s^{r_{str},r_{seq},k} = r_{str} \cdot A_{str}^{MSP} + r_{seq} \cdot A_{seq}^{MSP} - w^{r_{str},r_{seq},k}.$$

Then the alignment A which minimizes dist(A) is the alignment that maximizes the similarity sim$(A) = \sum_{r_{str},r_{seq},k} s_A^{r_{str},r_{seq},k} \lambda_A^{r_{str},r_{seq},k}$, and vice versa.

The resulting scoring scheme is depicted in Table 4.1. As discussed above for A^{MSP}, a good choice for A_{str}^{MSP} (resp. A_{seq}^{MSP}) is to use the maximal cost for edit operations involving only structural (resp. sequential) positions. Another possibility is to choose A_{str}^{MSP} such that the maximal weight for a single edge (namely $2A_{str}^{MSP}$) equals the maximal value allowed in *T-Coffee*. This is a reasonable choice if there is a high confidence in the structures selected for the sequences, and one wants to ensure that the structural positions are aligned.

Combining Several Structures

The previously described approach uses one given structure for each sequence, which could be for example an experimentally confirmed structure. Usually, the structure is not known and has to be computed by secondary structure prediction programs like *Mfold* [Zuker, 1994] or *RNAfold* [Hofacker, 2003]. Here, we are confronted with the problem that very often the real motif is not found in the minimum free energy structure, but in some sub-optimal structures.

A better strategy is to assign several structures to each sequence covering different possible folds of the sequence. To generate an ensemble of low energy structures, we have used the stochastic backtracking version of *RNAsubopt*

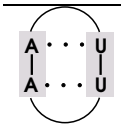
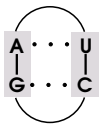
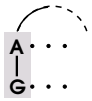
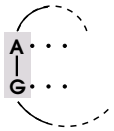
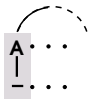
Edit-Op	Name	Distance	Similarity
	arc match	0	$4 \cdot A_{str}^{MSP}$
	arc mismatch	$w_{am}(A,U,G,C)$	$4 \cdot A_{str}^{MSP} - w_{am}(A,U,G,C)$
	arc breaking arc altering (realized edge)	$w_{base}(A,G) + \frac{1}{2}w_{ad}^{const}$	$A_{str}^{MSP} + A_{seq}^{MSP} - w_{base}(A,G) - \frac{1}{2}w_{ad}^{const}$
	arc breaking arc altering (realized edge, two arcs)	$w_{base}(A,G) + w_{ad}^{const}$	$2 \cdot A^{MSP} - w_{base}(A,G) - w_{ad}^{const}$
	arc breaking arc removing (non-realized edge)	$w_{base}(A,-) + \frac{1}{2}w_{ad}^{const}$	no realized edge

Table 4.1: Edit operations on arcs together with the associated distances and their similarity values given to the *T-Coffee*system. Note that for arc-match and arc-mismatch, we assign half of the total similarity value to each alignment edge when building the library.

(Vienna RNA Package) as well as *RNAshapes* [Giegerich et al., 2004]. The latter avoids the production of a large number of similar structures. The result is a usually small set of different structures $\mathcal{E}_S = \{P_S^1 \dots P_S^{n_S}\}$ for a sequence S. In the following, we call $\mathcal{E}_S$ the *ensemble* of the sequence S.

Since the structures P_S^k in $\mathcal{E}_S$ occur with different frequencies in the low energy spectrum, they have to be weighted. The weight for each structure is given by the conditional probability $\Pr(P_S^k|\mathcal{E}_S)$ of seeing this structure under the condition that only structures of the ensemble $\mathcal{E}_S$ are considered. Thus, we have

$$\Pr(P_S^k|\mathcal{E}_S) = \frac{\Pr^b[P_S^k]}{\sum_{1\leq l\leq n} \Pr^b[P_S^l]}, \tag{4.19}$$

where $\Pr^b[P_S^l]$ is the Boltzmann probability that S forms the structure P_S^l. Since *RNAshapes*often returns structures with similar energies, we approximate $\Pr(P_S^k|\mathcal{E}_S)$ by the uniform distribution in our current implementation of *MARNA*, thus avoiding the calculation of the Boltzmann probabilities.

Next, we have to use the different structures to form a single library for a pair of sequences. So let S_1 and S_2 be two sequences. Assume that we have selected n_1 structures for the first sequence and n_2 structures for the second one. In this setting, $n_1 = 1$ (resp. $n_2 = 1$) means that we have a unique known structure for S_1 (resp. S_2). Thus, we are able to mix sequences having known structures with sequences where we do not know the structures. Let $\mathcal{E}_{S_1} = \{P_{S_1}^1 \dots P_{S_1}^{n_1}\}$ and $\mathcal{E}_{S_2} = \{P_{S_2}^1 \dots P_{S_2}^{n_2}\}$ be the ensembles of structures selected for sequences S_1 and S_2, respectively. Then we perform $n_1 \times n_2$ many sequence structure alignments for $(S_1, P_{S_1}^k)$ and $(S_2, P_{S_2}^l)$ $(1 \leq k \leq n_1$, $1 \leq l \leq n_2)$. All realized edges from these alignments are then collected into a single library. For edges that are common to several alignments, the weights are summed up. In order to achieve weights that are consistent with other libraries, the combined similarity values of the realized edges are normalized by multiplying them by $\Pr(P_{S_1}^k \mid \mathcal{E}_{S_1}) \cdot \Pr(P_{S_2}^l \mid \mathcal{E}_{S_2})$.

4.3.4 Consensus Structure

Once we have computed the final alignment, we are ready to calculate a consensus structure from this alignment. The standard approach is to estimate possibly conserved bonds by means of the mutual information content (see e.g. Gutell and Woese [1990], Chiu and Kolodziejczak [1991], Gutell et al.

[1992]) between all columns i and j in a given alignment. The keynote is that if there is not very much sequence conservation in these columns, but the columns show a high correlation measured by the mutual information content, then this must be due to a conserved bond. Hofacker et al. [2002] extended this approach by considering the probabilities of forming these base-pairs.

In our case, the situation is different since we explicitly use structure information for the calculation of the alignment. Hence, the calculation of the consensus structure should be based on these given structures.

To exemplify the basic idea, suppose that exactly one structure per sequence is given. Thus, each structure must then be interpreted as the "real" known structure. A conserved base-pair between two columns in the alignment is found if the majority of sequences have a base-pair at the corresponding sequence positions. The remaining problem is that the resulting set of conserved base-pairs alone does not form a nested secondary structure and is thus not a valid consensus structure. This is a problem common to all approaches for calculating a consensus structure. The usual solution is to calculate a secondary structure that maximizes base-pair conservation. So let c, c' be two columns with $1 \leq c < c' \leq m$, where m is the number of columns of the multiple alignment. Furthermore, let $bp\_cons(c, c')$ be the number of sequences that have a base-pair between the corresponding sequence positions. The *consensus structure* is then defined to be a secondary structure $P \subseteq [1..m] \times [1..m]$ such that

$$\sum_{(c,c') \in P} bp\_cons(c, c')$$

is maximized. This can be calculated using a variant of the Nussinov algorithm [Nussinov et al., 1978, Luck et al., 1999]. For this purpose, we define a matrix $(N_{i,j})$ with $1 \leq i, j \leq m$, where

$$N_{i,j} = \max_{P} \sum_{\substack{(c,c') \in P \\ i \leq c < c' \leq j}} bp\_cons(c, c') \tag{4.20}$$

is the maximal base-pair conservation for all columns between i and j. The corresponding recursion equation for this matrix is

$$N_{i,j} = \max \begin{cases} N_{i+1,j}, \\ N_{i,j-1}, \\ N_{i+1,j-1} + bp\_cons(i,j), \\ \max\limits_{i<k<j} \{N_{i,i+k} + N_{i+k+1,j}\} \end{cases} \tag{4.21}$$

It is a dynamic programming approach, where the traceback reports the consensus structure of the alignment.

Finally, we have again to consider the case where we are given structure ensembles for some (or all) sequences. The overall structure of the approach is the same, only the definition of conserved base-pairs has to be adapted, i.e. the definition of $bp\_cons(c, c')$. If we have several structures for one sequence, then the probability of seeing a particular base-pair depends on the probabilities of the structures that contain this base-pair. Hence, we can only calculate the *expected number* of occurrences of base-pairs for two columns c and c'. Thus, we redefine $bp\_cons(c, c')$ as follows. Consider a multiple alignment of K sequences. For each sequence S_k, let $\mathcal{E}_k$ be the ensemble of structures calculated for S_k. For each column c, let i_c^k be either the position that corresponds to column c in sequence S_k (if aligned), or $-$ otherwise. Furthermore, let $\delta_P(c, c')$ be the index function of P, i.e. $\delta_P(c, c')$ is 1 if $(c, c') \in P$, and 0 otherwise. Then

$$bp\_con(c, c') = \sum_{k=1}^{K} \sum_{P \in \mathcal{E}_k} \delta_P(i_c^k, i_{c'}^k) \cdot \Pr[P \mid \mathcal{E}_{S_k}],$$

where $\Pr[P \mid \mathcal{E}_{S_k}]$ is defined as given in equation (4.19).

Complexity

Here, we assume that all sequences have nearly the same length L and that we generate an ensemble of E structures for each sequence. Note that by using *RNAshapes*, E is typically small (up to 3 sequences). The running time of one pairwise alignment is $O(E^2L^4)$. We have to make N(N-1)/2 comparisons in a set of N RNAs. Therefore, the pairwise comparison step needs $O(E^2N^2L^4)$ computation time. The most time consuming part of the multiple alignment step consists of building the extended library which takes

$O(N^3L^2)$ steps in the worst case [Notredame et al., 2000]. Altogether, the dominating alignment complexity is given by

$$O(E^2N^2L^4) + O(N^3L^2).$$

Chapter 5

Stable Sequence Structure Properties

Beside the comparison of sequence structure properties among two or more RNAs, another aspect concerning thermodynamic energy parameters characterizes an RNA structure. Sankoff [1985] already incorporates these energy parameters when folding and comparing multiple RNAs. Other approaches omit these energy contributions, even though these considerations are essential to exhibit stabilized conformations. In this chapter, we investigate stable RNAs by briefly recalling the minimum free energy(mfe) folding algorithm from Zuker and Stiegler [1981] (see also [Zuker, 1994]), implemented in the programs Mfold [Zuker, 2003] and RNAfold [Hofacker et al., 1994], as well as the partition function described by McCaskill [1990]. Here, we focus on locally stable RNAs by developing sophisticated algorithms to locate locally stable conformations in known as well as unknown secondary structures. We use energy values as published by Mathews et al. [1999]. For the ease of simplicity in the presentation, we don't consider additional energy parameters contributed by helix penalties and dangling ends.

Predicting the correct structure of an RNA often results in computing the minimum free energy structure using nearest neighbor thermodynamic rules. However, this method prevents considering the abundance of suboptimal structures including a vast amount of important local structure properties. In particular, locally stable conformations are mostly responsible for catalytic or regulatory functions in the cell. The evaluation or even the prediction of locally stable regions circumvent the computation of entire secondary structures and focus on stable regions providing more information

about e.g. RNA-DNA, RNA-RNA or RNA-Protein interactions.

In this chapter, we analyze locally stable regions in RNAs by proposing a computational scheme that uses the equilibrium partition function and the base-pair binding probabilities. Local structures can be evaluated by their probabilities in thermodynamic equilibrium. Two novel algorithmic approaches of predicting locally stable regions in RNAs with known as well as unknown secondary structures are presented.

We review the mfe folding algorithm as proposed by Zuker and Stiegler [1981] and the partition function as proposed by McCaskill [1990]. These two recurrence relations are needed to devise algorithms for predicting locally stable regions in known as well as unknown structures.

5.1 Globally Stable RNAs

The Zuker algorithm [Zuker and Stiegler, 1981], implemented in the program Mfold [Zuker, 2003] and the Vienna RNA package [Hofacker et al., 1994], identifies the mfe structure of an RNA sequence in an efficient way. It is a good starting point for structure analysis but may fail to correctly fold important local regions due to global sights. A first step to overcome these difficulties for genome-wide surveys was proposed by Hofacker and coworkers [Hofacker et al., 2004b]. They use a window sliding approach of size L computing the fold within this scope in a larger RNA or even in a complete genome. The folding structure can be seen as a sequence of contiguous mfe structures of small RNAs. Another problem that arises in optimal as well as in suboptimal foldings are RNA structure switches. Even if two structures of the same RNA have nearly the same energy, they might exhibit two completely different structural scaffolds. A major goal is to detect local regions that are shared by most of the suboptimal structures. Of course, if a certain local structure is known in advance, then motif search programs are capable to localize them.

The computation of the globally stable structure of an RNA is given by Zuker and Stiegler [1981]. It is one of the most popular and classic algorithms in computational biology when predicting secondary structures of RNAs. These recursion equations resemble the recursion equations for locally stable regions in RNAs which we will give later. Here, we briefly present the recursion equations for computing the mfe structure.

Recall the notions from section 2.2. The loop decomposition of secondary

structures lead to following recursion equations for computing the minimum free energy structure of an RNA. Five arrays V, W, VBI, VM and WM suffice to describe the recursions. In fact, the number of arrays can be reduced (preferably 3), but we will use five arrays to simplify the description. The free energy contributions $e(hp)$ (hairpin), $e(s)$ (stacked base-pairs), $e(bi)$ (loop closed by two base-pairs) and $e(ml)$ (multiple loop) are the contributions for various loops. Here, we subdivide the energy contribution $e(bis)$ into $e(s)$ for stacked base-pairs and $e(bi)$ for internal and bulged regions. Recall that the energy contribution for a multiple loop is decomposed into the linear equation $e(ml) = MA + m * MB + n * MC$, where MA, MB and MC are constants, m is the number of base-pairs within this loop and n is the number of unpaired bases. Then, the recursion equations for computing the mfe structure are given as:

$$W(i) = \min\{W(i-1), \min_{1<j\leq i}\{W(j-1)+V(j,i)\}\} \quad (5.1)$$

$$V(i,j) = \min\{e(hp(i,j)), e(s(i,j))+V(i+1,j-1), VBI(i,j), VM(i,j)\} \quad (5.2)$$

$$VBI(i,j) = \min_{\substack{i<i'<j'<j \\ i'-i+j-j'>2}}\{e(bi(i,j,i',j'))+V(i',j')\} \quad (5.3)$$

$$VM(i,j) = \min_{i+1<k\leq j-1}\{WM(i+1,k-1)+WM(k,j-1)+MA\} \quad (5.4)$$

$$WM(i,j) = \min\{V(i,j)+MB, WM(i,j-1)+MC, WM(i+1,j)+MC, \min_{i<k\leq j}\{WM(i,k-1)+WM(k,j)\}\} \quad (5.5)$$

- The array $W(i)$ is the energy of an optimal structure of the subsequence from 1 through i.
- The energy of an optimal structure of the subsequence from i through j closed by the base-pair (i,j) is given as $V(i,j)$.
- The energy of an optimal structure of the subsequence from i through j where the base-pair (i,j) closes a bulged or internal loop is given as $VBI(i,j)$.
- The energy of an optimal structure of the subsequence from i through j where the base-pair (i,j) closes a multiple loop is given in $VM(i,j)$.

- The energy of an optimal structure of the subsequence from i through j that constitutes part of a multiple loop is given in $WM(i,j)$. That is, where unpaired bases and external bases are penalized according to the linear loop function.

The recursion equations lead to a dynamic programming algorithm to calculate the optimal global structure in time proportional to n^3. The only difficulty to achieve this computation time is the array VBI. It requires as shown here $O(n^4)$ time, but the evaluation of bulged and internal loops to a constant size let the algorithm reduce to an $O(n^3)$ time algorithm. Further details can be found e.g. in [Lyngso et al., 1999].

5.2 Equilibrium Partition Function

McCaskill [1990] published a dynamic programming algorithm using time $O(n^3)$ and space $O(n^2)$ to calculate the full equilibrium partition function for secondary structures. Furthermore, this computational scheme allows the computation of the base-pair binding probabilities of an RNA of length n with the same complexity. Based on the calculation of the partition function, we demonstrate in the next subsection how to calculate the occurrence probabilities of arbitrary, local structures. First, we sketch the recursion equations of the partition function.

$Q_{i,j}$ denotes the partition function in the interval of positions between i and j. $Q^b_{i,j}$ denotes the partition function in the same interval on condition that base i and base j form a base-pair. $Q_{i,j}$ is the sum of Boltzmann weights over all secondary structures in thermodynamic equilibrium:

$$Q_{i,j} = \sum_{structure\, S} e^{-\frac{e(S)}{RT}} \tag{5.6}$$

where $e(S)$ is the energy of the secondary structure S. Consequently, the probability of a certain structure is then given as:

$$P(S) = \frac{1}{Q_{1,n}} e^{-\frac{e(S)}{RT}} \tag{5.7}$$

The partition function of an RNA sequence of length n is given as $Q_{1,n}$. It can be recursively computed:

$$
\begin{aligned}
Q^b_{i,j} &= e^{-e(hp(i,j))/RT} + \sum_{\substack{h,l \\ i<h<l<j}} e^{-e(bis(i,j,h,l))/RT} \\
&\quad + \sum_{\substack{h,l \\ i<h<j}} Q^m_{i+1,h-1} Q^{m1}_{h,j-1} e^{-(MA+MB)/RT} \qquad (5.8) \\
Q^m_{i,j} &= \sum_{i<h\leq j} (e^{-(h-i)MC/RT} + Q^m_{i,h-1}) \\
&\quad \times Q^b_{h,l} e^{-(MB+MC(j-l-1))/RT} \qquad (5.9) \\
Q^{m1}_{i,j} &= \sum_{l=i+1}^{j} Q^b_{i,l} e^{(j-l)MC/RT} \qquad (5.10) \\
Q_{i,j} &= 1 + \sum_{i\leq h\leq j} Q_{i,h-1} Q^1_{h,j} \qquad (5.11) \\
Q^1_{i,j} &= \sum_{i\leq l\leq j} Q^b_{i,l} \qquad (5.12)
\end{aligned}
$$

Initialization is given by $Q^1_{i,i} = 0$, $Q_{i,i} = 1$, $Q_{i+1,i} = 1$, $Q^b_{i,i} = 0$, $Q^m_{i,i} = 0$ and $Q^m_{i+1,i} = 0$. R is the gas constant and T is the absolute temperature ($T = 310.15K$). This recursion scheme becomes an $O(n^3)$ time algorithm by restricting the size of bis-loops to $u < u_m$ where u_m may be regarded as a constant, preferably $u = 30$. This algorithm proceeds from lower segments to larger segments in order to obtain the value $Q_{1,n}$.

Beside the calculation of occurrence probabilities for single structures and structure ensembles, one of the main application was to calculate the base-pair binding probabilities [McCaskill, 1990]. They are mostly visualized as dot plot matrices, the size of boxes correspond to the base-pair binding probabilities.

The occurrence probability of a base-pair (i, j), as the smallest entity of an RNA secondary structure, can be described by a sum of probabilities over all secondary structures containing this base-pair:

$$
P_{i,j} = \sum_{(i,j)\in structure\, S} P(S) \qquad (5.13)
$$

Entries of the matrix $P_{i,j}$, where i and j run from 1 to n, correspond to probabilities that base i is paired with base j. The recursion equation to

calculate these entries in reasonable time, i.e. $O(n^3)$, is given in [McCaskill, 1990].

5.3 Locally Stable Regions in RNAs

Here, we focus on computing locally stable regions in initially folded or unfolded RNAs. Basically, the equilibrium partition function and the base-pair binding probabilities published by McCaskill [1990] are used. We define local stability and give equations for occurrence probabilities of single structure elements. A structure element is one of the following loop types: hairpin, stack, right bulge, left bulge, internal loop and multi-loop (see section 2.2). Given a known secondary structure of an RNA, we are able to compute the probability of every partial structure. Furthermore, an algorithm is presented that reports the highest probable local structure, or, in other words, the most stable region of an RNA. Even if the secondary structure is not known, we propose a dynamic programming algorithm that computes a locally stable structure independent of a global structure. In general, this computational scheme facilitates the discovery of novel important regions of RNAs characterized through structural stability. The computation time is proportional to n^3 thus avoiding the intractable computation of exponentially many suboptimal structures. Our algorithms are implemented in the package *stableRNA* which is a versatile tool to analyze local regions of RNAs.

For a base-pair (i, j), we call the subsequence from $i + 1$ to $j - 1$ the inner part of the RNA. Energy parameters for different loops are listed in Mathews et al. [1999]. Helix penalties for helices ending with an A-U or G-U base-pair in a multi-loop or external loop (not enclosed by other loops) as well as dangling end contributions are not considered by our algorithm for simplicity of presentation. This makes it possible to use the partition function as published by McCaskill [1990]. The computation of the partition function with the involvement of helix penalties and dangling end contributions has been devised by Ding and Lawrence [2003], but this is out of interest here.

Local Structures

Now, we are ready to concentrate on local structures. Here, we define a local structure L to be a set of contiguous loops. Formally, a local structure L consists of k single loops (structure elements) such that

$$L = L_1 \cup L_2 \cup \ldots L_k \tag{5.14}$$

where a hairpin loop consists of a base-pair (i, j) and a single strand of nucleotides between $i+1$ and $j-1$, a bis-loop consists of two base-pairs (i, j) and (h, l), $i < h < l < j$, possibly with single-stranded nucleotides between position $i + 1$ and $h - 1$ *and* $l + 1$ and $j - 1$. When there are no single-stranded nucleotides on both sides, then this bis-loop consists of stacked base-pairs. A multiple loop consists of a closing base-pair (i, j) and (at least two) internal base-pairs (i_1, j_1), (i_2, j_2), ..., (i_k, j_k) possibly with single-stranded nucleotides between these base-pairs. This definition coincides with the definition given in section 2.2. Here, we emphasize that an internal base-pair in an RNA secondary structure belongs to two adjacent loops. For the definition of a local structure L, we further require that the single loops adjoin, i.e. each closing base-pair (i, j) of a single loop L_m also belongs to another loop L_n. The only exception to this is the outer loop. Note, that a local structure as defined here consists of loops, the smallest local structure is a single loop. This is in contrast to a single base-pair, which can be computed by recursion equations as defined by McCaskill [1990]. Any base-pair has a higher probability than its adjacent loop. And any single loop has a higher probability than a set of loops containing this single loop.

From a theoretic view, local structures can be defined as a decomposition of not necessarily adjacent single loops. For the purpose of occurrence probability computations, these regions can be calculated by means of complicated recurrence equations, but this is out of our interest. Moreover, our prediction algorithm becomes intractable.

Our definition is primarily intended to find stabilized base-paired regions. They need not necessarily sequentially connected. For instance, the smallest stable local structure is a single structure element consisting e.g. of stacked base-pairs (i, j) and (h, l) such that $i+1 = h < l = j-1$. The sequential part consisting of the bases positioned between $h + 1$ and $l - 1$ are disregarded for its declaration of the stable component. Here, we refer to structure elements because they constitute the whole or partial structural scaffold of an RNA. For each of them, a free energy contribution can be assigned, and thus makes it possible to compute the occurrence probabilities. We disregard the computation of occurrence probabilities of partial single stranded sequences (mostly used for target accesibility) explicitely since they have been already considered by Ding and Lawrence [2001]. Rather we focus on locally stable

structures occurring in many suboptimal structures which might be responsible for the exertion of their function due to e.g. binding or docking effects.

The loops are ideal to calculate the occurrence probability of a local structure L in thermodynamic equilibrium by considering all secondary structures S containing L:

$$P(L) = \sum_{L \subseteq structure\, S} P(S) \tag{5.15}$$

We emphasize that it is wrong to easily compute the product $\prod_{(i_l,j_l)\in L} P_{i_l,j_l}$ since entries in the base-pair matrix are computed independently. For instance, if we consider an internal loop composed of the base-pairs (i,j) and (h,l) such that $i < h < l < j$, then no other base-pair formation (m,n) with $i < m < h < l < n < j$ is allowed as long as we are interested in the probability of the internal loop formed by (i,j) and (h,l). However, the loop formed by (m,n) and (h,l) is counted in the base-pair binding probability calculation of (h,l) since all potential loops must be considered.

5.3.1 Probabilities of Single Structure Elements

We concentrate on three distinguishable structure elements, namely on hairpins (hp), on bis-loops consisting of stacks, bulges and internal loops, and on multi-loops. They describe loops involving one, two or more than two base-pairs, respectively. To calculate the probability of each structure element, let's start with hairpins. A hairpin is characterized through a base-pair (i,j) and a single strand of bases from position $i+1$ to $j-1$. The probability of (i,j) is given as $P_{i,j}$, meaning that this base-pair is involved in loops formed by base-pairs (h,l) such that $1 \leq h \leq i-1$ and $j+1 \leq l \leq n$. We know that a hairpin has energy $e(hp)$ and thus the Boltzmann weight $e^{-\frac{e(hp)}{RT}}$. The probability of this specific hairpin independent of the remaining bases positioned at $1,\ldots,i-1$ and $j+1\ldots,n$ is $e^{-\frac{e(hp)}{RT}}/Q^b_{i,j}$. Finally, the involvement of the base-pair probability relative to the entire sequence then yields the probability of this hairpin in the sequence:

$$P(hp) = P_{i,j}\frac{e^{-\frac{e(hp)}{RT}}}{Q^b_{i,j}} \tag{5.16}$$

This equation can be also written as $P(hp) = P(hp_{[i,j]}|(i,j))P_{i,j}$ to see the probability of forming the hairpin structure in the sequence by computing

the conditional probability of forming this hairpin in the interval $[i, \ldots, j]$ with (i, j) as a base-pair multiplied by the base-pair probability $P_{i,j}$.

Similarly, the energy contribution $e(bis)$ of a bis-element, where the base-pairs (i, j) and (h, l) with $i < h < l < j$ form this element, results in Boltzmann weight $e^{-\frac{e(bis)}{RT}}$. The sequence part $h + 1, \ldots, l - 1$ is free in adopting every admissible structure, or, in other words, in adopting every admissible structure in the interval $h, \ldots, l$ provided that (h, l) is a base-pair. Therefore, the probability of a bis-element is

$$P(bis) = P_{i,j} \frac{Q^b_{h,l}}{Q^b_{i,j}} e^{-\frac{e(bis)}{RT}} \tag{5.17}$$

One can easily derive the probability of a multi-loop with n radiating stems excluding the stem at base-pair (i, j):

$$P(ml) = P_{i,j} \frac{e^{-\frac{e(ml)}{RT}}}{Q^b_{i,j}} \prod_{(h_k, l_k)} Q^b_{h_k, l_k} \tag{5.18}$$

We are now able to calculate the probability of every single structure element. Note again, that it is a mistake to compute only the product of base-pairs involved in these structure elements.

5.3.2 Locally Stable Regions in Known Secondary Structures

Here, we tackle the problem of calculating the probabilities of all kinds of partial structures in a known secondary structure.

According to the definition of a partial structure given in 5.14, a stable, local structure consists of at least one single structure element. A single base-pair or a strand of unbound nucleotides are not considered to be stable while we derive our algorithm. But the incorporation of these two cases is not difficult since they are given in the base-pair probability matrix as $P_{i,j}$ entries or as the probabilities of target accessibility binding sites [Ding and Lawrence, 2001]. Furthermore, note that the binding probability of a base-pair (i, j) is always higher than the probability of forming a single structure element closed by (i, j) because $P_{i,j} \geq P_{i,j}T$ with $T \leq 1$ and each equation 5.16, 5.17 and 5.18 has the form $P_{i,j}T$.

We solve the problem of finding the most stable, partial structure. Probabilities of single structure elements can be determined by the formulas given above. Suppose that if we consider two adjacent loops that share a common base-pair (h, l), then we call a loop an inner loop if bases and base-pairs that belong to it are in the interval $h+1, \ldots, l-1$. The loop closed by base-pair (i, j) such that $i < h < l < j$ is called the outer loop. The occurrence probability of these two structure elements is therefore the probability of the base-pair (i, j) in the outer loop multiplied by the Boltzmann weight of both loop energies multiplied by the partition function of the inner 'wobbling' part. For instance, two adjacent stacks s_1 and s_2 with base-pair $(i+2, j-2)$ stacked on $(i+1, j-1)$ stacked on (i, j) have energy contribution $e_1 = e(bis((i+2, j-2), (i+1, j-1)))$ plus $e_2 = e(bis((i+1, j-1), (i, j)))$. The 'wobbling' part is determined by the interval $i+3, \ldots, j-3$. Thus, we can easily derive the occurrence probability of both stacks :

$$P(s_1, s_2) = P_{i,j} \frac{Q^b_{i+2,j-2}}{Q^b_{i,j}} e^{-\frac{e1+e2}{RT}} \tag{5.19}$$

Similarly, all kinds of adjacent loops may be considered. This computational scheme can be used for partial structures in a given or non-given secondary structure.

Algorithm

What is the most stable, local structure in a given conformation ? Of course, it is the local structure with the highest occurrence probability. This can be either a single structure element or a union of several contiguous structure elements. Algorithmically, we have to compute for every base-pair (i, j) the most stable, local structure starting from this base-pair and ending in the inner part such that either the whole subsequence from position i to j is computed to be stable, or there exists a set of base-pairs $S = \{(h_i, l_i) | i < h_i < l_i < j\}$ with $h_i < l_i < h_j < l_j$ for two base-pairs $(h_i, l_i), (h_j, l_j) \in S$ with $h_i < h_j$. The most important information to store is the Boltzmann weight of loop energies in an $n \times n$ matrix B^{max}. Since we know the entire structure in advance, there are at most $O(n)$ base-pairs and thus an array of size $O(n)$ suffices. For notational reasons according to the description of a base-pair as a tuple of two positions, we speak of a matrix. For a base-pair (i, j) an entry of $B^{max}(i, j)$ is determined by its corresponding loop. Either the current single loop closed by (i, j) starts a new stable, local structure or

an already stable, local structure at base-pair (h, l) is extended. Here, the base-pair (h, l) is the adjacent base-pair of (i, j) such that $i < h < l < j$. Additionally, the type of the current structure element has to be taken into account. $B^{max}(i, j)$ stores the Boltzmann weight of the energy of the newly found, local structure. Note that this local structure may consist of several contiguous loops, at least one. The probabilities themselves are stored in an extra matrix P^{max}. Probabilities of single structure elements are denoted as $P^s_{i,j}$ The pseudocode for computing the most stable region in a known RNA secondary structure is as follows:

for each base-pair (i, j)
  compute $P^s_{i,j}$ *of single loop closed by* (i, j)
  compute $B^{max}_{i,j}$ *of most stable, local structure starting with base-pair* (i, j)
  $P^{max}_{i,j} = P_{i,j} \frac{B^{max}_{i,j}}{Q^b_{i,j}}$, *the probability of the most stable, local structure.*

The for-loop is applied to base-pairs in an inside to outside manner, i.e. for two base-pairs (i, j) and (h, l) with $i < h < l < j$ the base-pair (h, l) is considered first. Probabilities and recurrence equations corresponding to this algorithm are listed in Table 5.1.

structure element	Probability $P^s_{i,j}$ of single structure element	max. weighted energy $B^{max}_{i,j}$	max. probability $P^{max}_{i,j}$
	$\frac{P_{i,j}}{Q^b_{i,j}} e^{-\frac{e(hp)}{RT}}$	$e^{-\frac{e(hp)}{RT}}$	$\frac{P_{i,j}}{Q^b_{i,j}} B^{max}_{i,j}$
	$P_{i,j} \frac{Q^b_{h,l}}{Q^b_{i,j}} e^{-\frac{e(bis)}{RT}}$	$e^{-\frac{e(bis)}{RT}} \begin{cases} B^{max}_{h,l} & ,if\ extendable \\ Q^b_{h,l} & ,otherwise \end{cases}$	$\frac{P_{i,j}}{Q^b_{i,j}} B^{max}_{i,j}$
	$\frac{P_{i,j}}{Q^b_{i,j}} e^{-\frac{e(ml)}{RT}} \prod_{(h_k,l_k)} Q^b_{h_k,l_k}$	$e^{-\frac{e(ml)}{RT}} \prod_{h_k,l_k} \begin{cases} B^{max}_{h_k,l_k} & ,if\ extendable \\ Q^b_{h_k,l_k} & ,otherwise \end{cases}$	$\frac{P_{i,j}}{Q^b_{i,j}} B^{max}_{i,j}$

Table 5.1: Recurrence relations of single structure elements and their extensions. Single structure elements are depicted in the left column: a hairpin (1st row), a bis-element (2nd row) consisting of a stack, bulge or internal loop and a multi-loop (3rd row). Probabilities of single structure elements are given in the second column. The partition function and the matrix of base-pair probabilities are pre-computed. Maximum weighted energies are given in the third column. The boolean term *extendable* is true, if $P_{i,j} e^{-\frac{e(bis)}{RT}} B^{max}_{h,l} / Q^b_{i,j} > P^{max}_{h,l} - t$ and $P_{i,j} e^{-\frac{e(bis)}{RT}} B^{max}_{h,l} / Q^b_{i,j} > P^s_{i,j} - t$, i.e. the probability of the extension to the current loop ($P_{i,j} e^{-\frac{e(bis)}{RT}} B^{max}_{h,l} / Q^b_{i,j}$) is compared to both the local structure, from which it should be extended ($P^{max}_{h,l} - t$) and to the current single structure element ($P^s_{i,j} - t$). The *extendable* term in the multi-loop row has to be adapted with appropriate indices. See text for further explanations. The last column shows the best probabilities starting at (i, j).

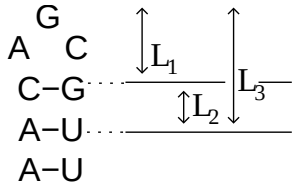


Figure 5.1: Example of a structure element extension in a stem-loop structure. L_1 is a hairpin with occurrence probability P_1, L_2 is a base-pair stack with probability P_2. If the inequalities 5.20 and 5.21 hold, then L_1 and L_2 are unified to local structure L_3.

Accepting small losses

In practice, often a single structure element of minimum free energy is chosen rather than a chain of contiguous structure elements. This crucial effect is due to the fact that loops adjoin such that an decreasing occurrence probability is expected while extending local regions. $Q^b_{h,l}$ has always a higher value than $B^{max}_{h,l}$ (see Table 5.1). Note here that $B^{max}_{h,l}$ contains the Boltzmann weight of at least one single structure element.

In order to make the computational scheme more flexible, we introduce a parameter t that accepts a small loss of probability values while extending local structures. For instance, consider the case of computing the extension of a local structure L_1 to a loop L_2 closed by base-pair (i, j) with internal base-pair (h, l). The probability of L_1 is given as P_1, the probability of L_2 as P_2 and the probability of both (L_3) as P_3. Then we want to extend L_1 to L_2, if $P_3 > P_1$ and $P_3 > P_2$ (see Figure 5.1). Since the occurrence probability of a small partial structure has always a higher value than a partial structure including this small partial structure, this extension cannot be fulfilled. Instead, we accept the extension if the following inequalities hold:

$$P_3 > P_1 - t \tag{5.20}$$

$$P_3 > P_2 - t \tag{5.21}$$

Indeed, for each inequality we need two parameters t_1 and t_2, respectively, where t_1 is the accepted loss of extending L_1 to L_3 when comparing the probabilities of L_1 and L_3. t_2 is the accepted loss when comparing the probabilities of L_2 and L_3. To simplify the extension process, we set $t = t_1 = t_2$. An extra

data structure $L_{i,j}$ is needed which stores the maximum value of $P_1 - P_3$ and $P_2 - P_3$, if both inequalities $P_1 - t < P_3 < P_1$ and $P_2 - t < P_3 < P_2$ hold. Otherwise $L_{i,j}$ is set to the value $P_1 - P_3$ (resp. $P_2 - P_3$), if $P_1 - t < P_3 < P_1$ (resp. $P_2 - t < P_3 < P_2$) holds. If none of the inequalities is valid, then $L_{i,j}$ is set to zero. The differences are the real loss of probabilities. If the loss has been accepted, then the old loss value, i.e. the error value of the local structure that has been extended (for a bis-element: $L_{h,l}$), has to be added. If the loss is too strong, then the local structure is not extended anymore. An example is shown in Figure 5.2.

Finally, the correct structure can be obtained by finding the maximum value of

$$P_{i,j}^{max} + L_{i,j}$$

and starting a traceback procedure. Once the right starting base-pair has been found, we ignore all remaining $L_{i,j}$ values since the correct Boltzmann weights are stored in B^{max}. The traceback is easily performed on Boltzmann weights of energy parameters and not on probabilities.

The parameter t is mainly used for bis-elements. If we consider hairpins, then we don't need this parameter since hairpins don't enclose any other structure element. In a multi-loop, there are more than two internal base-pairs. The extension step is accomplished by considering each internal base-pair separately.

Here, we make use of the parameters t_1 and t_2, which seem to be not very well-suited for adjusting. Since its justification is given by the loss of probabilites, i.e. rather by their mathematical interpretation than by their biological interpretation, other parameters are useful to incorporate such as the size of the stable region. Here, we focus on the mathematical description.

5.3.3 Locally Stable Regions in an RNA Ensemble

Given a known structure, we presented an algorithm for the detection of the most stable local structure by investigating base-pairs and their incident loops in an inside to outside manner. The number of base-pairs is restricted to $O(n)$. In an RNA ensemble, i.e. without a known secondary structure, we need to consider all potential base-pairs and all types of their incident loops. The number of base-pairs is restricted to $O(n^2)$. Here, we propose an $O(n^3)$ dynamic programming approach to obtain the most stable local structure in an RNA ensemble. It is similar to the mfe algorithm from Zuker and Stiegler

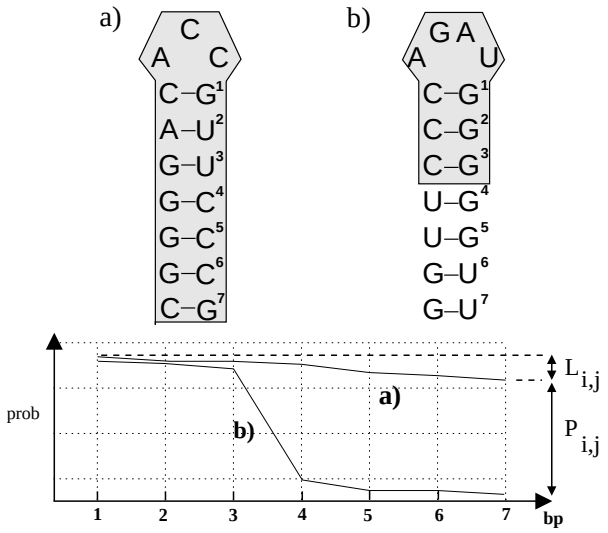


Figure 5.2: Illustration of local probabilities in terms of two stem loop structures. Both consists of seven stacked base-pairs. The algorithm proceeds from inside to outside, i.e. the probabilities of the hairpins are considered first. In general, a loss of probability is expected while extending the local structure. The loss of probability in the first RNA is acceptable as long as the inequalities 5.20 and 5.21 still hold. The loss is stored in matrix L, whereas the real probability is stored in the matrix P^{max}. In the second RNA, the loss of extending the third base-pair to the fourth base-pair is too strong. The two stacks and the hairpin are indicated as stable.

[1981]. Three matrices of size n^2 each are used to keep values of Boltzmann weights at positions (i, j) of potential base-pairs

1. enclosing the most stable region found so far (matrix B),
2. assuming that this current base-pair is an internal base-pair in a multi-loop (matrix B^m) and
3. assuming that this base-pair is external, i.e. not included in any loop (matrix C^m).

The difference between B^m and C^m is in the computation of Boltzmann weights caused by different energy contributions. An external base-pair has no energy contribution, whereas a base-pair in a multi-loop determines the

energy contribution according to the linear equation $e(ml) = MA+m*MB+n*MC$. The parameter t is used again with the same meaning as in the last section. Again, a small loss of occurrence probability in the recurrence equations is permitted. This tolerance 'error' is stored in an extra matrix $L_{i,j}$ such that the starting point of the most stable local structure is

$$\max_{i,j} \{F_{i,j} + L_{i,j}\} \tag{5.22}$$

As we can see, we assume that a local structure begins with an external base-pair (i, j). Otherwise, $F_{i,j}$ has to be replaced by $B_{i,j}$. Since we don't know if a locally stable structure starts with an external or internal base-pair, we simply assume the base-pair to be external. Matrix $L_{i,j}$ is initialized with zero values. Matrix entries are filled with values derived from the recurrence equations as shown in Figure 5.3.

The time complexity of the algorithm shown in Figure 5.3 is $O(n^4)$ because of running two variables h, l between the values i and j in $B_{i,j}$. Again, we apply the standard approach to reduce the time complexity from $O(n^4)$ to $O(n^3)$ by restricting the size of $l - h$ to a constant u such that $l - h \leq u$ ($u = 30$).

The most stable local structure can be obtained by a traceback step comparing Boltzmann weights of corresponding loops rather than probabilities.

Our algorithm is formally similar to the mfe algorithm except that we use Boltzmann weights instead of energies, we take maximum values instead of minimum values and values are multiplied instead of added.

$$
\begin{aligned}
B_{i,j} = \max \quad \{ \quad & e^{-\frac{e(hp)}{RT}}, \\
& e^{-\frac{e(bis)}{RT}} \max_{i<h<l<j} \left\{ \begin{array}{ll} B_{h,l} & ,if\, extendable \\ Q^b_{h,l} & ,otherwise \end{array} \right. , \\
& \max_{i<k<j} \{ e^{-\frac{MA+MB}{RT}} B^M_{i+1,k} B^M_{k+1,j-1} \} \} \\
B^M_{i,j} = \max \quad \{ \quad & e^{-\frac{MB}{RT}} \left\{ \begin{array}{ll} B_{i,j} & ,if\, extendable \\ Q^b_{i,j} & ,otherwise \end{array} \right. , \\
& e^{-\frac{MC}{RT}} B^M_{i+1,j}, e^{-\frac{MC}{RT}} B^M_{i,j-1}, \\
& \max_{i<k<j} \{ B^M_{i,k} B^M_{k+1,j} \} \} \\
C^M_{i,j} = \max \quad \{ \quad & \left\{ \begin{array}{ll} B_{i,j} & ,if\, extendable \\ Q^b_{i,j} & ,otherwise \end{array} \right. , \\
& C^M_{i+1,j}, C^M_{i,j-1}, \\
& \max_{i<k<j} \{ C^M_{i,k} C^M_{k+1,j} \} \} \\
F_{i,j} = \max \quad \{ \quad & \frac{P_{i,j}}{Q^b_{i,j}} B_{i,j}, \frac{C^M(i,j)}{Q_{i,j}} \}
\end{aligned}
$$

Figure 5.3: Recursion equations for detecting the most stable local RNA. Since local structures are defined as chains of contiguous loops, all recursion equations are applied to base-pairs in a dynamic programming fashion, i.e. the computation proceeds from smaller to larger segments. Recursion on multi-loops are marked as the superscript M. B^M considers closed multi-loops, whereas C^M considers external multi-loops, i.e. base-pairs are non-enclosed. The effect is in the energy calculation. The maximum value of $F_{i,j}$ determines the end of a local structure. From this position, the traceback procedure reports the most stable local structure. It is ingenious to employ a tolerance parameter accepting a small loss of probabilities during the extension step as given in the inequalities 5.20 and 5.21. This is included in the boolean term *extendable*, which true if $P_{i,j} e^{-\frac{e(bis)}{RT}} B^{max}_{h,l} / Q^b_{i,j} > P^{max}_{h,l} - t$ and $P_{i,j} e^{-\frac{e(bis)}{RT}} B^{max}_{h,l} / Q^b_{i,j} > P^s_{i,j} - t$. t is the tolerance parameter. h and l are must be exchanged with i and j where necessary. All matrices are initialized with zero values.

Chapter 6

Results

In this chapter, we present numerous examples that were computed on the basis of the described, theoretical concepts in this thesis. These implementations comprise the detection of maximally extended patterns common to two RNA secondary structures (described in section 3.2.2), the multiple alignment tool *MARNA*(described in section 4.3), both in the field of sequence structure comparisons, and the locally stable detection program *stableRNA*(described in chapter 5) in the field of stable regions of RNAs.

Exact Pattern Matching

The first program detects exact, maximally extended patterns common to two RNA secondary structures. The program is written in C++. The output is a sorted list of patterns according to their descending sizes. The sorting algorithm dominates the running time, but it can be omitted if desired. Recall that the time complexity is $O(nm)$ and the space complexity is also $O(nm)$. The maximal number of patterns is at most a fraction of nm. Therefore, the expected time complexity of the sorting algorithm is $O(nmlog(nm))$ (worst case: $O(n^2m^2)$) when listing all patterns in descending order.

MARNA

MARNA is the abbreviation for **M**ultiple **A**lignment of **RNA**s. *MARNA* is a web-server reachable under the address
http://www.bioinf.uni-freiburg.de/Software/MARNA/index.html.
and also available as a downloadable source code.

MARNA is a multiple alignment method for RNAs considering both the primary sequences and the secondary structures. It generates pairwise sequence structure alignments and combines them using *T-Coffee*. Hence, *MARNA* is not only a structure alignment tool, but also considers sequence similarities. The main advantage is to set individual parameter values capable of fine-tuning sequence- and structure preferences. Concerning structures, one can use either user-defined structures, or let *MARNA* predict an ensemble of low energy structures. A simple example demonstrating the relevance of structural properties is given in Figure 6.3.

MARNA can be tested online on our website. Although a pairwise comparison needs time complexity of $O(n^2m^2)$ for two RNAs of lengths n and m, and thus limits the input sequence lengths to about 500 bases, *MARNA* has been tested successfully on many RNAs like tRNAs, rRNAs, ncRNAs etc.

In this chapter we go through the results with a fine-tooth comb. We propose a contribution score to score multiple alignments and compare them with stochastic context-free grammar methods (SCFG) such as *Cove* [Eddy and Durbin, 1994] , *RNACAD* [Brown, 1999b] and *PMmulti* [Hofacker et al., 2004a]. Furthermore, consensus -sequences and -structures are predicted and compared with *PMmulti* and *RNAalifold* (part of the Vienna RNA Package).

stableRNA

stableRNA is a program written in $C++$ which computes locally stable regions in initially folded or unfolded RNAs. This program is threefold. It is able to compute a local structure in thermodynamic equilibrium, to predict a local structure with max. probability given a global structure and to predict a local structure with max. probability in thermodynamic equilibrium. This program can be used, among other things, to detect locally stable regions in genomic sequence regions.

6.1 Exact Pattern Matching

The algorithm of detecting exact, maximally extendend patterns in two RNA secondary structures is implemented in C++. The output is a sorted list of patterns according to their descending sizes. The sorting algorithm dominates the running time, but it can be omitted if desired.

We have performed two tests showing that i) although global structures of RNAs diverge, they share a lot of local sequence structure properties and ii)

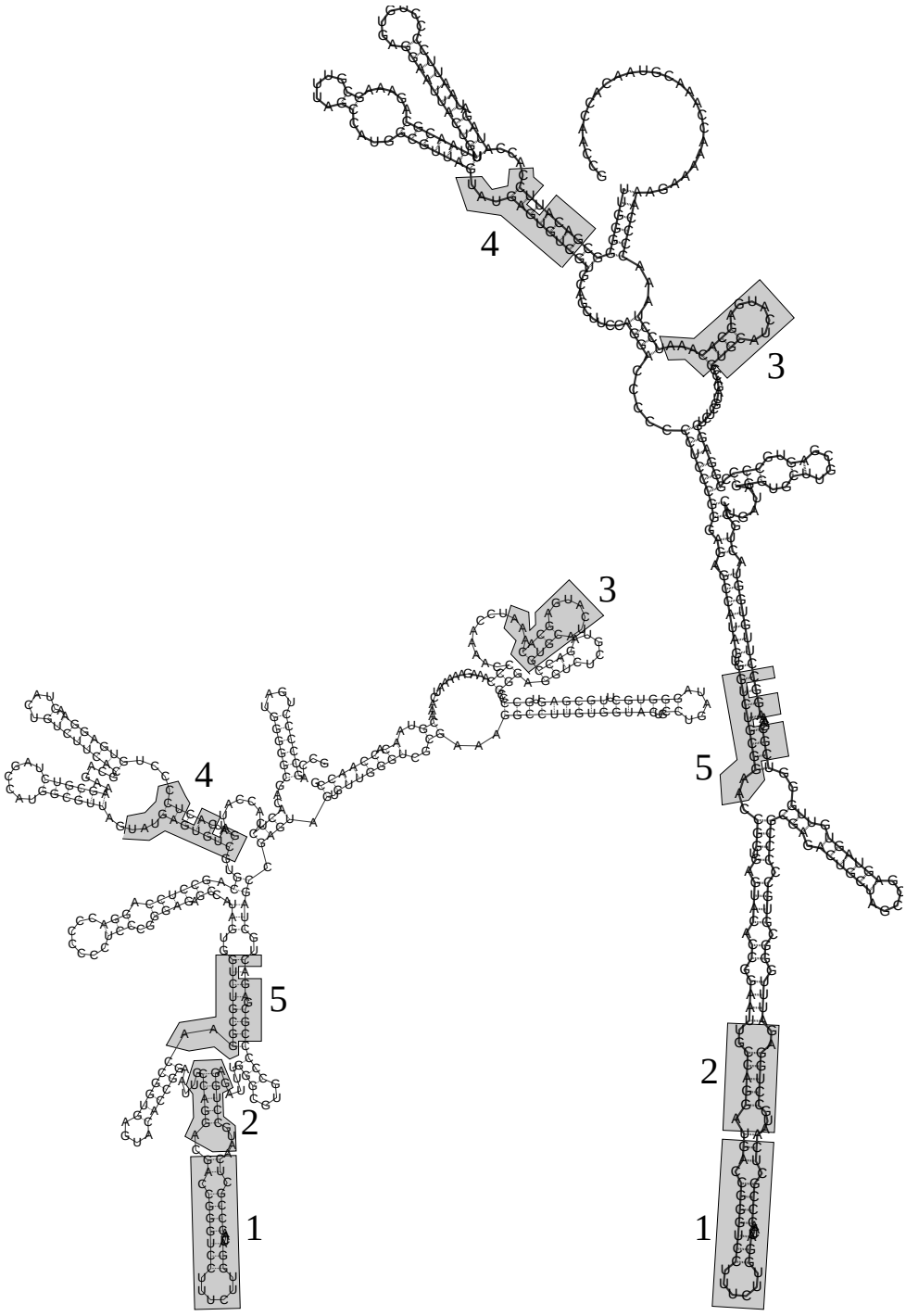


Figure 6.1: Hepatitis C virus internal ribosome entry sites (IRES) of two RNAs. They are given by their EMBL accession codes: D45172 (left RNA) and AF165050 (right RNA). The five largest patterns in terms of their sequential and structural properties are highlighted. The ascending numbering of patterns is equivalent to the descending size of patterns. The pattern numbered 1 is the maximum common pattern.

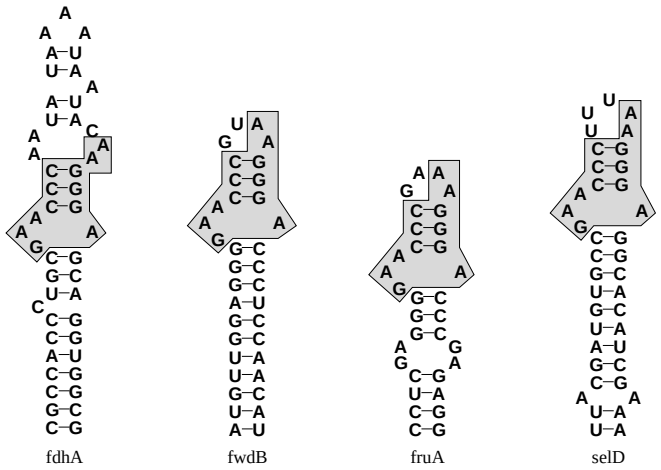


Figure 6.2: Putative SECIS-elements in non-coding regions of *Methanococcus jannaschii*. The algorithm detects a strongly conserved pattern by pairwise comparisons. This pattern was also discovered by hand from Wilting et al. [1997].

strong conserved regions of several RNAs can be easily detected by comparing them pairwise.

a)IRES : The first test is performed on Hepatitis C virus internal ribosome entry sites (IRES). The RNAs are taken from the RNaseP Database [Brown, 1999a] given by their EMBL accession codes: AF165050 and D45172. Their sequence lengths are 391 and 379, respectively. We have folded both RNA sequences into their energetically optimal structures with the use of *RNAfold* from the Vienna RNA Package [Hofacker et al., 1994]. The two RNAs are shown in Figure 6.1. At first sight, their global structures differ enormously. Our common pattern detection algorithm exhibits big similarities in both RNAs. We have highlighted the five largest common patterns. We can see that both RNAs share a lot of sequential and structural patterns although we have taken only the computed minimum free energy structures. The size of the maximum common pattern is 30. Altogether, the algorithm has found 8829 patterns.

b)SECIS: The second test was performed on putative SECIS-elements (Selenocysteine Insertion Sequence) in non-coding regions of *Methanococcus jannaschii*. They were taken from Wilting et al. [1997]. By pairwise compar-

ison we yielded a strongly conserved region which coincides with the results found manually by Wilting et al. [1997]. These patterns are the maximally sized patterns.

6.2 MARNA

6.2.1 Contribution Score

This subsection deals with the score definition of a single RNA contributing to a *MARNA* alignment.

Each RNA can be evaluated by its contribution score. It indicates how strong the influence of each RNA is to the multiple alignment. Since each RNA is compared to each other, an amount of realized alignment edges is generated. However, they do not necessarily appear in the multiple alignment again. Fixing a certain RNA in a multiple alignment, the contribution score of that RNA is the sum of weights assigned to realized alignment edges (starting from this RNA) divided by the total weight of the entire alignment. Formally, we define the weight of an entire multiple alignment A as:

$$w(A) = \sum_{\substack{e \text{ is realized} \\ \text{alignment edge}}} w(e)$$

The weighted contribution of one RNA R to the entire alignment is then given as:

$$w(R) = \frac{1}{w(A)} \sum_{\substack{e \text{ is realized alignment} \\ \text{edge starting in } R}} w(e)$$

We also call it the *contribution score* of R. Note that if we sum up the contribution scores of each RNA then we achieve a score which is twice as high as the total score of the multiple alignment. This is due to the fact that each alignment edge is counted twice, namely from each end of the edge.

Test data

We have tested several programs dealing with secondary structures. They all rely on initial multiple alignments which are given by diverse multiple

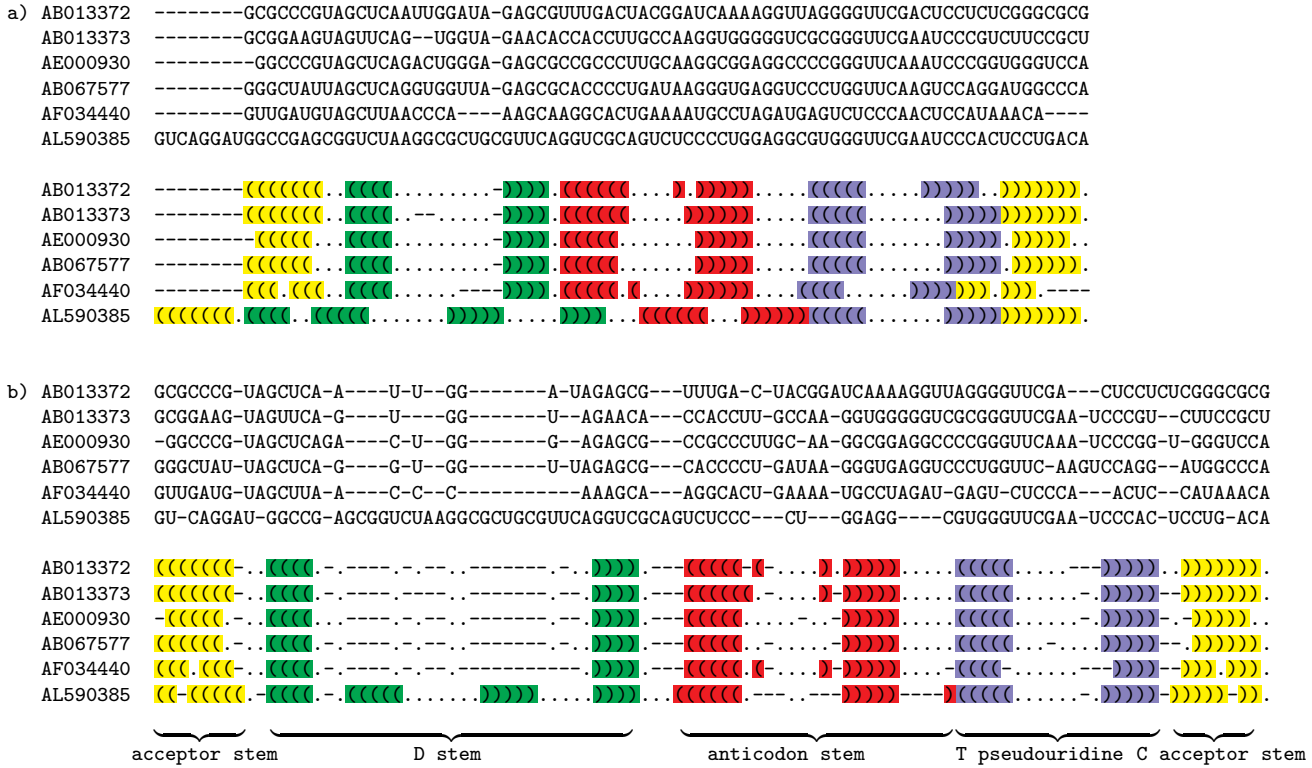


Figure 6.3: Alignments of tRNA sequences performed by *ClustalW* and *MARNA*. Base-pairs are colored due to stem affiliation. a) *ClustalW*: Structures are aligned according to sequence alignment. There is no clear consensus structure observable. b) *MARNA*: All stem loops are aligned according to their characterizing tRNA stems.

Test sets	Sequences
A) tmRNAs	acti.acti, haem.infl, kleb.pneu, past.mult, salm.para, salm.typh, shew.putr, vibr.chol, yers.pest
B) bacterial SRP RNAs	bac.alc, bac.bre, bac.cer, bac.cir, bac.mac, bac.meg,bac.pol, bac.pum, bac.sph, bac.ste, bac.thu, bre.bre, clo.per
C) eukaryotic SRP RNAs	ory.sa-a, tri.ae-a, tri.ae-b, zea.ma-a, zea.ma-b, zea.ma-c, zea.ma-d, zea.ma-e, zea.ma-f, zea.ma-h

Table 6.1: Test sets as used by Knudsen and Hein [2003]. They are taken from Zwieb et al. [2003] and Rosenblad et al. [2003].

alignment tools, among *MARNA* belongs to. The output of these programs can be evaluated by their own scoring functions and thus evaluate *MARNA* alignments indirectly. We have chosen three well-known methods based on probabilistic theory using SCFG: *Pfold* [Knudsen and Hein, 2003], *Cove* [Eddy and Durbin, 1994] and *RNACAD* [Brown, 1999b]. Furthermore, we extended our method by incorporating the optimal as well as suboptimal structures of primary sequences and compared our results to *PMmulti* [Hofacker et al., 2004a]. Here, we obtained nearly the same results as *PMmulti* does, whereas we could improve all results in case of the SCFG methods. We have used the following parameter values for all *MARNA* alignments:

$$
\begin{aligned}
w^a_{mismatch} &= 1.5 & \qquad w^b_{mismatch} &= 1\\
w^a_{altering} &= 1.75 & \qquad w^b_{deletion} &= 1\\
w^a_{replacing} &= 1 & &\\
w^a_{removing} &= 2 & &
\end{aligned}
$$

a)*Pfold*: *Pfold* [Knudsen and Hein, 2003] is a web server for structure prediction of RNAs computing a common secondary structure. The input is an alignment without explicit structural information. *Pfold* relies on related

Test set	Manual	ClustalW	MARNA
A)	95.8 %	96.5 %	96.6 %
B)	95.4 %	89.1 %	94.2 %
C)	92.3 %	88.6 %	90.9 %

Table 6.2: The common structure of each test set is predicted with *Pfold*. Three different alignments are used as input. The accuracy function as shown in per cent exhibits reliable, good results in all three test sets using *MARNA* alignments.

sequences. Therefore, we decided to take the same published data sets shown in Table 6.1. We have taken three different kinds of alignments as input:

A) manual alignments provided by the databases,

B) multiple sequence alignments computed by *ClustalW* and

C) our *MARNA* alignments with structures provided by the databases.

To compare the results we used a prediction accuracy function similar to the one given in [Knudsen and Hein, 2003]. The accuracy is the percentage of positions for which each structure is the same as the predicted common structure. A binding base is only counted if the whole base-pair is aligned correctly. Here, we assume that each RNA has its 'correct' structure from the databases.

The results of the accuracy computation are shown in Table 6.2. The test sets are chosen such that the sequences are quite closely related. This might be the reason that the accuracy values in case of *ClustalW* alignments are relatively high; the accuracy value is even better than the value obtained by the manual alignment of the test set A. A slightly better result for the same test is achieved by *MARNA*. For test sets B and C, one can observe that *MARNA* alignments are competitive with the manual alignments presuming that manual alignments are 'correct'. *MARNA* alignments exhibit reliable, good results in all test cases.

b) *Cove*: Eddy and Durbin [Eddy and Durbin, 1994] wrote a covariance model(CM) program reflecting primary sequences and secondary structures of RNA families in a probabilistic manner. It is adaptive which means that a model can be learned from trained sequences.

For our comparison approach we have chosen 26 human SECIS elements taken from Kryukov et al. [2003]. Structures of the primary sequences can be obtained by SECISearch 2.1, a program for identifying and folding SECIS elements that fulfill certain binding and non-binding constraints. SECISearch is available online at http://genome.unl.edu/SECISearch.html. In addition, a manual alignment of all SECIS elements is given in the supplement to their paper.

Cove is able to generate a covariance model either from trusted alignments or from unaligned sequences. In our test cases, trusted alignments are given by the manual alignment, the *ClustalW* alignment and our *MARNA* alignment. Sequences can be scored by the program *Cove*(see Table 6.3).

The human SECIS elements are an interesting test case because they are related less closely than the data sets given in Table 6.2a. Therefore, it is not surprising that covariance models from *ClustalW* alignments are badly constructed. But even worse is the CM construction from unaligned sequences. The scorings of the sequences show that in case of the *MARNA* alignment the construction of the CM is quite well (mean value=15.27), although it does not reach the results based on the manual alignment (mean value=19.83).

c)*RNACAD*: *RNACAD* is a stochastic context-free grammar (SCFG) RNA modeling package that accounts for both primary and secondary structure information. For our purpose, we used the same data set of SECIS elements and the same alignments as in b). This package needs an alignment and a representative structure for generating a model. The representative structure is given by 15kDa in all test cases. A model is computed that reflects the secondary structure and the statistics of the sequences. Sequences are scored by means of model construction (Table 6.4).

It is not surprising that the model construction fails in using the *ClustalW* alignment. More interesting are the results using the *MARNA* alignment. They are very close to the results obtained by the manual alignment.

d)*PMmulti*: *PMcomp* and *PMmulti* are two programs to perform pairwise and progressive multiple alignments of RNA sequences [Hofacker et al., 2004a]. *PMmulti* is a variant of Sankoff's algorithm for simultaneous folding and alignment of multiple sequences. The data set is taken from their paper showing the IRES Ib region from Aphthovirus and Cardiovirus.

We modified *MARNA* to predict the three topmost energetically favored structures to each RNA and performed the multiple alignment step. One can think of the existence of each RNA three times, each with a different structure. The running time increased ninefold due to the pairwise compar-

SECIS elements	Manual	ClustalW	unaligned	MARNA	MARNA scores
SelP14	20.95	-2.75	4.60	15.93	4.01 %
SelP18	2.99	-5.78	2.81	1.41	3.80 %
SPS2	17.53	15.47	8.30	18.81	3.98 %
SelW	16.92	13.06	5.85	15.41	3.89 %
SelV	33.29	11.56	5.67	20.25	3.95 %
15kDa	18.04	8.98	-4.85	19.60	4.07 %
SelM	22.12	-2.80	11.21	9.34	3.98 %
TR1	14.91	3.57	-8.22	7.59	3.57 %
TR2	20.54	2.86	-5.10	11.76	3.48 %
TR3	28.03	15.23	-1.25	22.52	3.94 %
GPx1	5.59	7.55	1.96	1.84	3.96 %
GPx2	22.19	5.38	10.83	17.36	3.93 %
GPx3	10.40	8.88	8.97	14.09	3.83 %
GPx4	26.56	11.96	10.16	24.25	3.38 %
GPx6	17.23	9.46	6.55	15.30	3.67 %
DI1	17.97	5.63	1.86	15.44	3.78 %
DI2	23.74	10.51	-10.65	14.82	4.06 %
DI3	16.46	5.41	10.90	10.81	3.89 %
SelR	31.43	5.26	5.19	17.72	4.15 %
SelT	26.38	9.93	-4.95	23.18	3.74 %
SelN	23.60	0.33	9.86	14.40	3.66 %
SelH	10.89	4.83	4.92	14.47	3.81 %
SelK	20.93	-4.53	0.00	18.40	3.73 %
SelS	19.43	-2.70	3.00	15.59	3.91 %
SelI	25.12	11.87	2.21	10.43	3.93 %
SelO	22.37	-10.50	1.41	26.28	3.88 %
min	2.99	-10.50	-10.65	1.41	3.38 %
max	33.29	15.47	11.21	26.28	4.15 %
mean	19.83	5.33	3.12	15.27	3.85 %

Table 6.3: Scoring of all 26 human SECIS elements in bits using covariance model version 2.4.4 (*Cove*). For each alignment, a covariance model is constructed and sequences are scored against this model. SECIS elements are not very sequentially related which explains the bad scoring values using the *ClustalW* alignment. Even the CM construction of unaligned sequences is worse. *MARNA* provides reasonable good results which are by far better than those generated by the *ClustalW* alignment or the unaligned sequences.

SECIS elements	Manual	Clustalw	MARNA	MARNA scores
SelP14	26.95	12.63	24.32	4.01 %
SelP18	9.21	8.58	17.78	3.80 %
SPS2	26.51	22.48	24.23	3.98 %
SelW	37.00	34.94	37.19	3.89 %
SelV	47.30	7.66	40.50	3.95 %
15kDa	37.97	22.86	37.54	4.07 %
SelM	24.50	17.44	4.40	3.98 %
TR1	21.04	17.79	28.09	3.57 %
TR2	28.87	20.83	30.87	3.48 %
TR3	41.68	25.62	29.26	3.94 %
GPx1	21.21	16.86	24.05	3.96 %
GPx2	31.57	23.17	23.69	3.93 %
GPx3	19.89	31.42	24.92	3.83 %
GPx4	36.30	14.18	34.95	3.38 %
GPx6	25.41	27.45	35.32	3.67 %
DI1	25.69	12.96	30.74	3.78 %
DI2	25.63	18.13	24.21	4.06 %
DI3	31.27	17.23	34.09	3.89 %
SelR	41.34	17.49	39.88	4.15 %
SelT	31.27	22.00	35.16	3.74 %
SelN	36.48	13.05	33.46	3.66 %
SelH	30.31	4.50	28.98	3.81 %
SelK	21.45	18.40	27.67	3.73 %
SelS	21.83	25.32	29.07	3.91 %
SelI	21.18	17.57	31.23	3.93 %
SelO	39.20	13.32	17.60	3.88 %
min	9.21	4.50	4.40	3.38 %
max	47.30	34.94	40.50	4.15 %
mean	29.27	18.61	28.82	3.85 %

Table 6.4: Scoring of SECIS elements using *RNACAD*. The test set as well as the alignments are the same as for *Cove*. Here, the scoring values using the *MARNA* alignment are close to the values of the manual alignment. The last column shows the contribution scores of *MARNA* in percent. The scores are quite consistent whereas the scores of *RNACAD* diverge between 4.4 and 40.5.

ison step. The resulting structure-enhanced multiple sequence alignment is used for the computation of the consensus structure graph by *RNAalifold* [Hofacker et al., 2002], part of the Vienna RNA Package. The results are competitive with the results of the recently published method *PMmulti* [Hofacker et al., 2004a]. Results are given in the next section; they are depicted in Figure 6.7.

6.2.2 Consensus Structure

In this section, we tested several multiple alignment tools to derive consensus structures from multiple alignments (see section 4.3.4).

The simple example shown in Figure 6.3 consists of six tRNAs taken from the RNA families database of alignments Rfam [Griffiths-Jones et al., 2003]. Each tRNA is chosen randomly and are approximately 80 nucleotides in length (accession codes: AB013372, AB013373, AE000930, AB067577, AF034440, AL590385). They are folded into their optimal structures using *RNAfold* from the Vienna RNA Package. Parentheses denote base-pairs. They all show the characteristic cloverleaf structure.

The output of *MARNA* is shown in Figure 6.3a. In fact, all sequence and structure aspects are recognized and aligned correctly. It is clear, that structural diversity of the RNAs affects the multiple alignment of *MARNA*.

The running time depends, on the one hand, on the lengths of the sequences and, on the other hand, on the number of sequences. To get a rough insight, the tRNA example needs about 10 seconds. If the length of each RNA is doubled, but the number is consistent, the alignment is computed in about 55 seconds. If also the number is doubled, *MARNA* needs about 3 minutes and 55 seconds (data not shown). The machine used here is a Pentium IV, 2.4GH, 1 GB RAM, with SuSE Linux 8.1.

In addition, we compared our method with the traditional multiple sequence alignment program *ClustalW* [Thompson et al., 1994]. It is characterized by only producing sequence alignments, i.e. ignoring structures. Using default values the result is shown in Figure 6.3b. Structures are also shown which are aligned according to the sequence alignment.

Test data

The tRNA example demonstrates the importance of considering structural properties. To achieve a consensus structure, we illustrate comparable

methods based on three data sets from the Rfam database [Griffiths-Jones et al., 2003]. We compared the alignments as well as the consensus structures as provided by the Rfam database with the output of the two different alignment tools *T-Coffee* and *PMmulti*. The manual alignment and the consensus structure from the Rfam database serve as the reference. For *MARNA* and *PMmulti*, we have compared the consensus structures as proposed by the programs. If *RNAalifold* [Hofacker et al., 2002] computes a more representative consensus structure, we have also displayed this one. For *T-Coffee*, we have used *RNAalifold* [Hofacker et al., 2002] to predict the consensus structure.

a)SECIS: The first data set consists of seven randomly chosen eukaryotic SECIS-elements (Rfam accession number RF00031). SECIS-elements are necessary for the incorporation of selenocysteine into a protein sequence directed by an in-frame UGA codon (usually a stop codon) within the coding region of the mRNA. Selenoprotein mRNAs contain a conserved secondary structure in the 3' UTR that is required for the distinction of UGA stop from UGA selenocysteine. The sequences are around 60 nt in length and adopt a hairpin structure which is sufficiently well-defined and conserved, but the primary sequences differ. This data set is especially hard for sequence structure alignment programs since it contains 4 non-standard base-pairs (U–U, G–A, A–G, C–U) in the lower part of the stem.

The manual alignment was made from 25 SECIS-elements out of a total set of currently 65 SECIS-elements. We have chosen 7 out of the 25 sequences randomly. The manual alignment is shown in Figure 6.4a and serves as the 'true' alignment. The alignment computed by *T-Coffee* reveals some nucleotide similarities among sequences, as expected, but are not suited for the prediction of a consensus structure. Here, *MARNA* detects the long stem structure with the characteristic bulged A's in the upper loop [Lambert et al., 2002]. For this test case, we used the predicted minimum free energy structure for each sequence.

We also aligned all 65 SECIS-elements using *MARNA* with mfe structures and with structure ensembles calculated by *RNAshapes* and by stochastic backtracking [Ding and Lawrence, 2003] implemented as the option '-p' in the program *RNAsubopt*(Vienna RNA Package). We have compared these results with the results of *PMmulti*. The results are summarized in Table 6.5.

b) tRNA-like structures: The second data set consists of 22 tRNA-like structures, found in the 3' UTR of Tymoviruses and Pomoviruses. They were also taken from the Rfam database. The family is thought to be involved in the initiation of minus-strand synthesis and the disruption of the pseudoknot

gives rise to a 50% drop in transcription efficiency. *MARNA* (both with mfe structures as well as with structure ensembles) is able to detect the four stems as well as the single G between the first and second stem. For *PMmulti*, the four stems are detected when using *RNAalifold* afterwards (see Figure 6.7).

Finally, we have used the objective function given by Bali Base benchmark program [Thompson et al., 1999] to compare all generated alignments, again taking the Rfam alignments as a reference. The benchmark program returns two scores, namely SP (sum of pairs) and TC (total columns). SP measures the ratio of the number of correctly aligned pairs, whereas TC measures the number of correctly aligned columns. Since conservation of columns is different in sequence alignments and sequence structure alignments, we have used only the SP-score. The results are summarized in Table 6.5.

6.2.3 Choosing the Right Structures

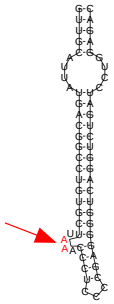
MARNA allows the user to assign different kinds of structures to each sequence. Beside the specification of user-defined structures, *MARNA* permits the assignment of mfe structures [Zuker and Stiegler, 1981], shaped structures as described by Giegerich et al. [2004] and highly probable structures using stochastic backtracking [Ding and Lawrence, 2003]. The last case allows the user to assign a concrete number of structures to each sequence. The default value is set to three.

Depending on the structure choice, the display of multiple structures are combined on single rows consisting of stars. A row with stars characterizes the ambiguous assignment of multiple structures to a sequence. The consensus sequence is successively constructed column by column. A capital letter indicates the full conservation of a nucleotide. A lowercase letter indicates a nucleotide conservation in at least 50% of all sequences. The consensus structures reflect the conservation of base-pairs in per cent occurring in all structures.

As an example case, consider the hairpin ribozyme. The hairpin ribozyme, like the hammerhead (RFAM:RF00008 and RFAM:RF00163), is found in RNA satellites of plant viruses. It was first identified in the minus strand of the tobacco ringspot virus (TRSV) satellite RNA where it catalysis a self-cleavage reaction to process the products of rolling circle virus replication to unit-length satellite RNA. The structure consists of 2 domains, A and B. The 3' arm of domain A, which closes helices 1 and 2, contains the cleavage site and is linked to helix 3 of domain B by a linker of variable length

a)Manual Alignment :

```
L37762     CUCGCUAUAUGACGAUGGCAAUC.UCAA..AUGUU....CAUU...GGUUGCCAUUUGAU.GAAAUCAGUUUUGUGUG
U67171     GACGCUUCAUGAUAGGAAGGACU.GAAA.AGUCUU.GUGGACACCUGGUCUUUCCCUGAU.GUUCUC......GUGGC
AB022283   GCCAGAUGAUGAGGACCUGUGCG.GAAA.CCCCCC.G.CGGGC...UGCCCAUGUCUGAG..CCC........CUGGC
X12367     GUUUUUCCAUGACGGUGUUUCCUCUAAA..UUUAC....AUG....GAGAAACACCUGAUUUCCAG......AAAAAU
AL049837   GUGUGCGGAUGAUAACUACUGAC.GAAA.GAGUCAU.CGACUC..AGUUAGUGGUUGGAU..GUAG......UCACAU
AF136399   GUCAGAUGAUGAUGGCCUGGGCA.GAAA..CCCCAUG.UGGGC...CGCCCAGGUUUGAA..CCC........CUGGC
S79854     CACUGCUGAUGACGAACUAUCUC.UAA.CUGGUCUUG..ACCA..CGAGCUAGUUCUGAA..UU.G.......CAGGG
```



b) *T-Coffee* Alignment :

```
L37762     CUCGCUAUAUGACGAUG---GCAAUCUCAAAUGUUCAUUG-----------GUUGCCAUUUGAUGAAAUCAGUUUUGUGUG
U67171     GACGCUUCAUGAUAGGA--AGGACUGAAAAGU----CUUG-UGGACACCUGGUCUUUCCCUGAUGUUCUCGUGGC------
AB022283   GCCAGAUGAUGAGGAC-----CUGUGCGGAAACCCCCC---------GCGGGCUGCCCAUGUCUG------AGCCCCUGGC
X12367     GUUUUUCCAUGACGGUGUUUCCUCU----AAAUUUACAUG----------GAGAAACACCUGAUUUCCAGAAAAAU-----
AL049837   GUGUGCGGAUGAU---AACUAC--UGACGAAAGAGUCAU----CGACUCAGUUAGUGGUUGGAUGUAGUCACAU-------
AF136399   GUCAGAUGAUGAUGGC-----CUGGGCAGAAA-CCCCAUG-----UGGGCCGCCCAGGUUUGAA---------CCCCUGGC
S79854     CACUGCUGAUGACGAA-----CUAUCUCUAACUGGUCUUGACCACGAGCUAGUUCUGAAUUGCAGGG--------------
```

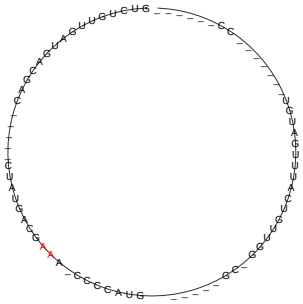


Figure 6.4: Comparison of multiple alignments and consensus structure predictions of seven randomly chosen SECIS-elements taken from the Rfam database. a) The restriction of the manual alignment in Rfam to the seven sequences. The consensus structure is proposed by Rfam, i.e. generated of all 65 SECISelements. b) The *T-Coffee* alignment is based on nucleotide similarities and thus disregards structural conformations. It is not surprising that the consensus structure contains no base-pairing interactions.

c)*PMmulti* Alignment :

```
L37762    -CUCGCU-AUA-UGACGAUGGCAAUCUCAAAUGUUCAUU-GGUUGCCAUUUGAUGAAAUCAGUUUUGUGUG
U67171    GACGCUUCAUGAUAGGAAGGACUGAAAAGUCUUGUGGACACCUGGUCUUUCCCUG-AUG-UUCU-CGUGGC
AB022283  ---GCC--AGAUGAU-G-AGG-A-CCUGUGCGGAAACCCCCCGCGGGCUGCCCAUGUCUGAG-CCCCUGGC
X12367    ---GUU--UUUCCAUGA-CGGUGUUUCCUCUAAAUUUACAUGGAGAAACACCUGAUUUC-CA-G-AAAAAU
AL049837  ---GUG--UGCGGAUGA-UAACUACUGACGAAAGAGUCAUCGACUCAGUUAGUGGUUGGAUGUAGUCACAU
AF136399  ---GUC--AGAUGAUGA-UGGCCUGGGCAGAAACCCCAUG-UGG-GCC-GCCCAGGUUUGAA-CCCCUGGC
S79854    ---CAC--UGCUGAUGA-CGAACUAUCUCUAACUGGUCUU-GAC-CACGAGCUAGUUCUGAA-UUGCAGGG

L37762    -..(((.-(((-.((((((((((....((.......))-..)))))))).(.....)..))).))))))
U67171    ...(((..(((..((((((((((....((.......))...)))))))).(..-..)-.)))-))))))
AB022283  ---(((--((..(.(-(-.((-(-(((.((.((...)...))).))).).)))(....)..)-)..)))))
X12367    ---(((--((..(.(.(-.((.(.(((.((.((...)...))).))).).)))(....)-.)-)-.)))))
AL049837  ---(.(--((....(..-.....................................)....))).)
AF136399  ---(.(--((..(.(.(-.((((((.((.(....(((...-.))-)).-)).)))))))..)-.).))).)
S79854    ---(.(--((..(.(.(-.((((((.((.(....(((...-.))-)).)).)))))))..)-.).))).)
```

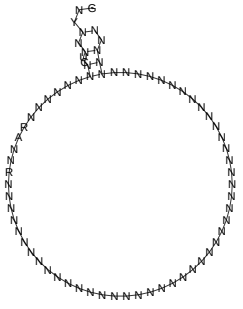


Figure 6.5: c) *PMmulti* detects some bonds, but does not identify the hairpin.

d) *MARNA*:

```
L37762    CUCGCUA---U-AUGA-CGAUGGCAA-UCUCAA-----AUGU-UCA-UUGG-UUGCCAUUU-GA-UGAAAUCAGUUUUGU-GUG-
U67171    GA---CG-CUUCAUGAUAGGAAGGAC-U-GAAAAGUC-UUGUGGACACCUG-GUCUUUCCCUGA-UGUUCU--------CGUGGC
AB022283  G------CCAG-AUGA--UGAGGACC-UGU-GC-GGA-AACCC-CC-CGCGGGCUGCCCAU-GUCUGAGCC--------CCUGGC
X12367    GU--UUUUCC--AUGA-CGGUGUUUCCUCUAAA-----UUUA--C-AUGGA-GAAACACCU-GA-UUUC--CA----GAAAAAU-
AL049837  GU--GUG-CGG-AUGA-UAACUA-CUGACGAAAGAGUCAUC-GACUC-AGUUAG-UGGUUG-GA---UG--UA----GUCACAU-
AF136399  GU--CAG-AUG-AUGA-UGGCCU-GGGCAGAAACCCC-AUGU-GGGC-CG-CCC-AGGUUU-GA---AC--CC------CUGGC-
S79854    CA--C-UGCUG-AUGA-CGAACUAUC-UCUAACUGGU-CUUG-ACCA-CGA-GCUAGUUCU-GA---AU--U-----GCA-GGG-

L37762    ..(((.(---(-(.((-(((((((((-((....-----....-...-..))-)))))))).-..-........))).))))-)))-
U67171    .(---((-...(((.(..(((((((((-(-.....(((-.....)))....)-)))))))).).)-)).....--------)))...
AB022283  (------((((-((.(--((.((.((-(((-(.-((.-.....-))-))))))...)))))-)))))).)).--------......
X12367    ((--(((((.--..((-(((((((((.((((..-----....--.-.))))-)))))))).-).-))..--..----)))))))-
AL049837  ((--(((-(..-....-((((((-(((((....(((((...-)))))-.)))))-))))))-..---..--..----).)))))-
AF136399  ((--(((-...-....-.(((((-((((.(....(((-....-))))-.)-)))-)))))).-..---..--..------)))))-
S79854    ..--(-(((.(-((..-.((((((.(-((....((((-....-))))-.))-).))))))).-..---))--)-----)))-)..-
```

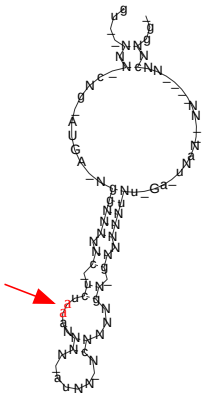
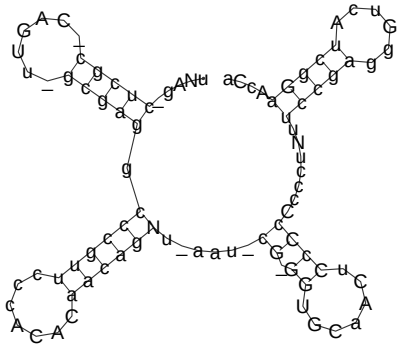
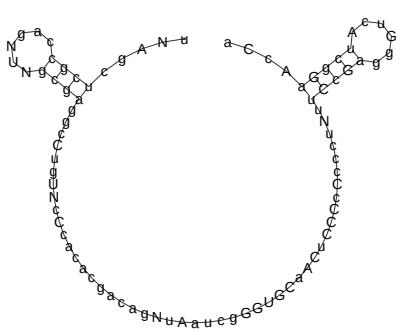


Figure 6.6: d) *MARNA* finds the hairpin-like structure as well as the bulged A's in the upper loop of the motif (indicated by a red arrow), which are required for the SECIS-element.

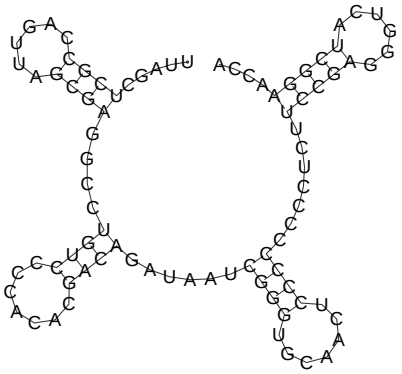
a) Manual Alignment



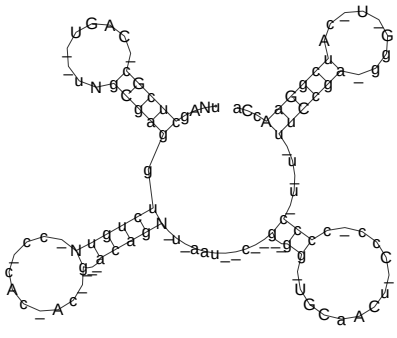
b) PMmulti



c) PMmulti+RNAalifold



d) MARNA(mfe)



e) MARNA(shapes)

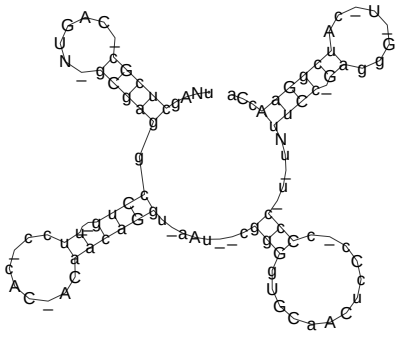


Figure 6.7: Consensus structure predictions for 22 tRNA-like structures, found in Tymovirus and Pomovirus. a) The consensus structure of the manual alignment contains four distinct stems. b) The proposed structure of *PMmulti* detects two of the four stems. c) An improvement of the structure prediction applied to the same computed alignment can be achieved by *RNAalifold* (part of the Vienna RNA Package). d,e) *MARNA* is able to detect all four stems in case of assigning the mfe structure as well as the shape structure to each sequence. Both consensus structures are very close to the Rfam consensus structure.

Sequences	**RFam Acc.-Number**	**MARNA (mfe)**	**MARNA (shapes)**	**MARNA (ens)**	**PMmulti**
SECIS-elements (7 rand.)	RF00031	0.327	0.351	0.545	0.286
SECIS-elements (all 65)	RF00031	0.463	0.487	0.447	0.162
Tymovirus/ Pomovirus	RF00233	0.715	0.782	0.837	0.730
Hammerhead	RF00008	0.785	0.811	0.742	0.696

Table 6.5: Evaluation of *MARNA* and *PMmulti* alignments using the SP-score of the Bali Base benchmark program. The first and third data set have been already analyzed in Figures 6.6 and 6.7. Additionally, we compared the whole data set of the 65 SECIS-elements and the Hammerhead ribozymes (type III). *MARNA* has been tested in different combinations, namely with mfe structures and structure ensembles generated by *RNAshapes* and by the stochastic backtracking of *RNAsubopt* (using 3 structures per sequence). Values in the table indicate the similarity to the reference manual alignment. They vary between 0 and 1. The maximum value for each test set is highlighted. They are reached by *MARNA* alignment with shape and ensemble structures.

```
D00721  CAACAGCGAAGCGCGCCAGGGAAACACACCAUGUGUGGUAUAUUAUCUGGCA
M21212  CAACAGCGAAGCGGAACGGCGAAACACACCUUGUGUGGUAUAUUACCCGUUG
M14879  AAACAGAGAAGUCAACCAGAGAAACACACGUUGUG..GUAUAUUACCUGGUA
D00685  CAACAGCGAAGCGCGCCAGGGAAACACACCAUGUGUGGUAUAUUAUCUGGCA
M17439  AAACAGAGAAGUCAACCAGAGAAACACACGUUGUG..GUAUAUUACCUGGUA
Str     .............(((((.......(((...)))..........)))))).
```

Figure 6.8: Alignment of hairpin ribozymes as given in the Rfam database. The seed alignment which serves as the training set for the whole alignment consists of the sequences M21212, D00685 and M17439. This alignment construction reveals a consensus structure made of two little stems.

```
D00721     CAACAGCGAAGCGCGCCAGG-GAAACACACCAUGUGUGGUAUAUUAUCUGGCA-
M21212     CAACAGCGAAGCGGAAC-GGCGAAACACACCUUGUGUGGUAUAUUACCCG-UUG
M14879     AAACAGAGAAGUCAACCAGA-GAAACACA-CGU-UGUGGUAUAUUACCUGGUA-
D00685     CAACAGCGAAGCGCGCCAGG-GAAACACACCAUGUGUGGUAUAUUAUCUGGCA-
M17439     AAACAGAGAAGUCAACCAGA-GAAACACA-CGU-UGUGGUAUAUUACCUGGUA-
D00721     .............((((((-....(((((...)))))........)))))).-
M21212     ...(((((..((.....-.))....(((((...)))))..........))-)))
M14879     .............(((((.-.((.(((.-...-.))).....))..)))))).-
D00685     .............((((((-....(((((...)))))........)))))).-
M17439     .............(((((.-.((.(((.-...-.))).....))..)))))).-
Seq        cAACAGcGAAGcgNacCaGg_GAAACACAcCNUgUGUGGUAUAUUAcCuGgua_
Str(30%)   .............((((((_.((.(((((...))))).....)).)))))).._
Str(40%)   .............((((((_.((.(((((...))))).....)).)))))).._
Str(50%)   .............(((((._....(((((...)))))..........))))).._
Str(60%)   .............(((((._....(((((...)))))..........))))).._
Str(70%)   .............(((((._....(((.......)))..........))))).._
Str(80%)   .............(((((._....(((.......)))..........))))).._
Str(90%)   ..................._....(((.......)))................._
Str(100%)  ..................._....(((.......)))................._
```

Figure 6.9: Alignment constructed by *MARNA* with mfe structures. The consensus sequence is nearly optimal in identifying equal bases. This alignment also depends on the provided mfe structures. Here, all sequences share the main stem with its upper stem loop. Only the sequence M21212 contains an additional stem in the first half of the sequence.

```
D00721    CAACAGCGAAGCGCGCCAGG-GAAACACACCAUGUGUGGUAUAUUAUCUGGCA----
M21212    CAACAGCGAAGCGGAAC-GGCGAAACACACCUUGUGUGGUAUAUUA-C---CCGUUG
M14879    AAACAGAGAAGUCAACCAGA-GAAACACA-CGU-UGUGGUAUAUUACCUGGUA----
D00685    CAACAGCGAAGCGCGCCAGG-GAAACACACCAUGUGUGGUAUAUUAUCUGGCA----
M17439    AAACAGAGAAGUCAACCAGA-GAAACACA-CGU-UGUGGUAUAUUACCUGGUA----
D00721    ********************-**********************************----
M21212    ...(((((..((.....-.)).....(((((...))))).........-.---.)))))
M14879    ..............(((((.-.((.(((.-...-.))).....))..))))).-----
D00685    ********************-**********************************----
M17439    ..............(((((.-.((.(((.-...-.))).....))..))))).----
Seq       cAACAGcGAAGcgNacCaGg_GAAACACAcCNUgUGUGGUAUAUUANCuggca____
Str(30%)  .....(((...)))((((((_.((.((((((...)))))).....)).))))))._____
Str(40%)  .....(((...)))((((((_.((.((((((...)))))).....)).))))))._____
Str(50%)  ..............(((((._....(((((...))))).........))))).____
Str(60%)  ..............(((((._....(((((...))))).........))))).____
Str(70%)  ..............(((((._....(((.......)))..........))))).____
Str(80%)  ..............(((((._....(((.......)))..........))))).____
Str(90%)  ..................._....(((.......)))..............____
Str(100%) ..................._....(((.......)))..............____
```

Figure 6.10: Alignment constructed by *MARNA* with shaped structures using *RNAshapes* [Giegerich et al., 2004]. A row with stars indicates an ambiguous assignment of multiple structures to a sequence (D00721,D00685). The remaining specified structures agree with the mfe structures in this case. The consensus structures of up to 40% conservation contain a little stem loop which must be due to the shaping structure construction. This is the primary difference compared to the consensus structures detected by the mfe assignment.

```
D00721     CAACAGCGAAG-CGCGCCAGG-GAAACACACCAUGUGUGGUAUAUUAUCUGGCA-
M21212     CAACAGCGAAGCGGAAC--GGCGAAACACACCUUGUGUGGUAUAUUACCCG-UUG
M14879     AAACAGAGAAG-UCAACCAGA-GAAACACA-CGU-UGUGGUAUAUUACCUGGUA-
D00685     CAACAGCGAAG-CGCGCCAGG-GAAACACACCAUGUGUGGUAUAUUAUCUGGCA-
M17439     AAACAGAGAAG-UCAACCAGA-GAAACACA-CGU-UGUGGUAUAUUACCUGGUA-
D00721     ***********-*********-***********************************-
M21212     ****************--***********************************-***
M14879     ***********-*********-*********-***-*******************-
D00685     ***********-*********-***********************************-
M17439     ***********-*********-*********-***-*******************-
Seq        cAACAGcGAAG_NgaaCcaGg_GAAACACAcCNUgUGUGGUAUAUUAcCuGgua_
Str(30%)   .....(((...._)))(((((._....(((((...)))))(((...)))))))).._
Str(40%)   .....(((..._)))(((((._....(((((...)))))(((...)))))))).._
Str(50%)   ..........._...(((((._....(((((...)))))((.....)))))))._
Str(60%)   ..........._...(((((._....(((((...)))))((.....)))))))._
Str(70%)   ..........._...(((((._....((((.....))))).........)))))._
Str(80%)   ..........._...(((((._....((((.....))))).........)))))._
Str(90%)   ..........._........._....((((.....))))..............._
Str(100%)  ..........._........._....((((.....))))..............._
```

Figure 6.11: Alignment constructed by *MARNA* with highly probable structures using stochastic backtracking; implemented in *RNAsubopt* as part of the Vienna RNA Package. The number of structure assigned to each sequence is set to three. The consensus structures of up to 40% conservation contain the same little stem loop as already detected by the *MARNA* alignment with shaped structures. Additionally, a little stem loop in the last part of the sequences is detected in up to 60% of all structures.

and sequence. This region is thus not included in this family. The hairpin ribozyme has been identified in only 3 plant satellite RNAs – from TRSV, chicory yellow mottle virus (CYMV) and arabis mosaic virus (ARMV).

The alignment as well as the detection of a consensus sequences/-structure depends on the structure choice. The original alignment is given in Figure 6.8. All kinds of structure assignments and its alignment computation by *MARNA* have been tested. The mfe structure assignment is given in Figure 6.9. The alignment using shaped structures is depicted in Figure 6.10 and the alignment using highly probable structures is depicted in Figure 6.11. In all *MARNA* alignments, the main stem loop in the middle part of the sequences can be found everywhere. As long as the structure assignment becomes more ambiguous, the more structure dispersion is included. In Figure 6.10, a little stem loop structure has been detected in two sequences. Additionally, in Figure 6.11, a further little stem loop appears in three sequences.

6.3 Locally Stable RNAs

Our algorithms for the detection and computation of locally stable regions in RNAs are implemented in C++ and can be obtained by downloading the package *stableRNA* via the webpage http://www.bioinf.uni-freiburg.de/Software. The package provides mainly three possibilities to investigate local stability of RNAs. First, the probability of a particular local structure can be calculated. For instance, if an RNA exhibits several stem-loops, then probabilities of all stem loops can be calculated separately. This computation can also be easily done by using the program RNAfold with constraints. Second, if an entire secondary structure of an RNA is given, e.g. the mfe structure, *stableRNA* reports the most stable local structure. The parameter t can be set as an optional argument. Default value is $t = 0.1$. Third, *stableRNA* reports the most stable local structure in an ensemble of a primary RNA sequence in thermodynamic equilibrium. All three algorithms are tested extensively.

Test data

We present four examples of how to reveal important local structure information, especially in dependence of the tolerance parameter t. The first example (IRE element) demonstrates the detection of local regions with the differentiation of varying parameters t_1 (equation 5.20) and t_2 (equation 5.21),

both which are usually set to the parameter $t = t_1 = t_2$ for its manageable usage. Hence, based on this, the size of the local region may change (and thus the probability).

The second example computes probabilities of partial structures in a known secondary structure. This computation may be also computed by the program *RNAfold* using constraints as input. The last two examples are realistic examples, first, to predict and measure locally stable regions in human immunodefiency virus (primer binding site) and, second, to detect locally stable regions which are known to have a function (Histone 3' UTR stem-loop, Hepatitis C stem-loop).

a) IRE element: An iron response element (IRE) element with accession number AF117958 was taken from the Rfam database [Griffiths-Jones et al., 2003]. It is 30 nucleotides in length. Its structure is a short stem-loop. The IRE element is found in UTRs of various mRNAs whose products are involved in iron metabolism. For our test case, we disregard the function of this element, rather we are interested in detecting locally stable regions dependent on the two parameters t_1 and t_2. They describe the accepted loss of probabilities during the extension step. We predicted the locally stable regions by running each parameter t_1 and t_2 from 0 to 1 with step size 0.1. Some test cases are depicted in Figure 6.12. As long as one of the parameters has a value of zero (top row and left column in Figure 6.12) the predicted local structure consists of a single structure element with minimum free energy. This element is a stack which consists of the two base-pairs G-C and U-A. Consequently, it has a high probability of 0.72. When predicting locally stable regions while running the parameters towards the value one, we make an interesting observation. The higher the parameters the more similar are the locally predicted structures to the mfe structure. But this is not surprising since we allow all kinds of losses. Hence, the globally stable structure agrees in this case with the locally stable structure. It has a small probability of 0.06 relative to its sequence length. A graphical presentation in Figure 6.13 demonstrates the distribution of the sizes and the occurrence probabilities of local regions dependent on the parameters t_1 and t_2.

b) RRE element: The Rev response element (RRE) is encoded within the env region of Human immunodeficiency virus type 1 (HIV-1). It is approximately 340 nucleotides in length (position 1537 to 1876) of the sequence with EMBL accession code AJ286340. The RRE is necessary for the Rev function; it contains a high affinity site for Rev. Rev is an essential regulatory protein of HIV that binds an internal loop of the RRE, encouraging further

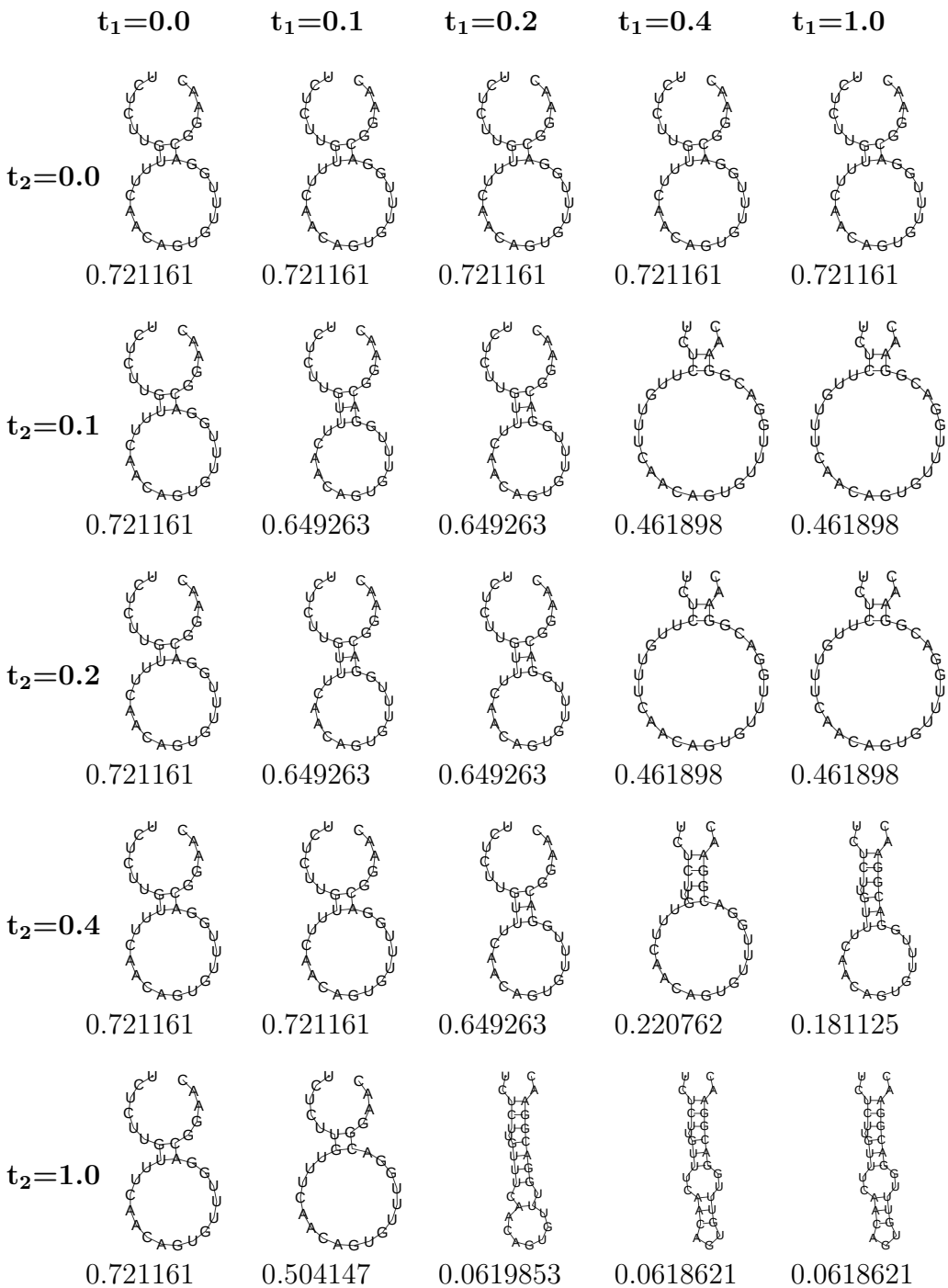


Figure 6.12: Prediction of the most stable local structures of a sample IRE element (acc. number AF117958, Rfam) in dependence of the two tolerance parameters t_1 and t_2. The top row and the leftmost column contain the most stable structure element that consists of a stack. Note that the surrounding bases, i.e. bases that are not involved in the stacking region are not determined by the structure. They might be involved in base-pairings. The higher the tolerance parameters the more similar are the locally predicted structures to the mfe structure. In this simple example, nearly all local structures occur in the mfe structure. This must not be necessarily the case, especially when considering larger RNAs.

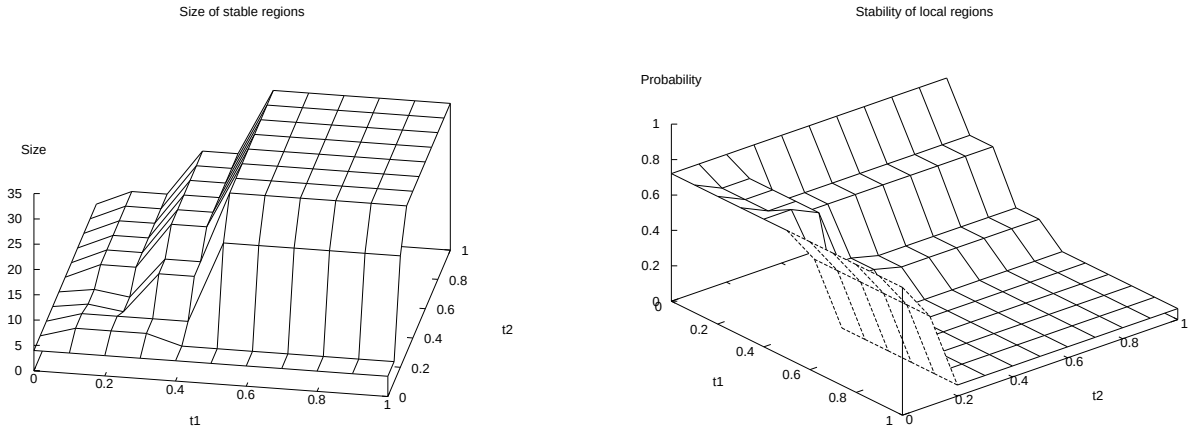


Figure 6.13: The size- and probability distribution of stable regions in the IRE element shown as 3d plots. The left figure depicts the sizes of the computed locally stable regions dependent on the two parameters t_1 and t_2. For $t_1 = 0$ or $t_2 = 0$, the size of the local region is four. It consists of the two stacked base-pairs. The more flexible the algorithm is in finding the locally stable region, i.e. the higher the values t_1 and t_2, the greater the size of the region. The higher plateau reaches a size of 30 which is the sequence length of the IRE element. The stable region thus consists of the whole sequence, i.e. the IRE element adopt the mfe structure. The right figure shows the probability distribution according to the parameters t_1 and t_2. Here, one can observe that the smaller the locally stable region the more stable are the regions. The lower plateau reflects the probability of adopting the mfe structure.

Rev-RRE binding. The RRE element is folded into its optimal structure of energy -161.40 kcal/mol yielding a frequency of $5.9 * 10^{-5}$ in the ensemble. It is almost equal to the conserved consensus structure given in the Rfam database. The structure can be divided into seven partial structures (see Figure 6.14). Our stability program reveals important results concerning these structural regions. Although stem I seems to be weakly stabilized, the probability of 0.01 is relatively high if one takes into account the number of approximately 180 nucleotides. However, this stem is less stable than the remaining stems II-V. Here, we observe incredible highly stable stem-loops. Even stem-loop IIc with the smallest occurrence probability(0.39) compared to stems II-V is highly stable. Stem II has been experimentally proven to contain a high affinity Rev binding site. To support this hypothesis, we analyzed this partial sequence by predicting the locally stable region in this partial sequence. The stem IIa has been predicted again with a probability of 0.95. Furthermore, the prediction of the stable regions in the partial sequence shown as stems IIb and IIc in Figure 6.14 reports again the stable stem IIb with probability 0.991 and the stem IIc with probability 0.998.

c) PBS element: Initiation of HIV-1 reverse transcription occurs by extension of the cellular $tRNA_3^{Lys}$ which anneals to the primer-binding site (PBS) on the 5' non-translated region of the viral RNA genome. The A-rich sequence (A-loop) upstream of the PBS interacts with the anticodon loop of $tRNA_3^{Lys}$. The proposed structure of the PBS domain is taken from Paillart et al. [2004] and shown in Figure 6.15a. It is dissected into the three stems I,II and III, whereas stem I can be further dissected into two stems. The A-loop can be found in stem II. The frequency of this structure is 0.000003, less stable than the computed optimal structure with a frequency of 0.05 (Figure 6.15b). The probabilities of all stems in both structures are computed. Stem II (Figure 6.15a) has a very low frequency, whereas stem IV (Figure 6.15b) has a moderate frequency of 0.56 with the A-loop given as a hairpin structure. Both structures contain the same stem I, which has been also predicted by our program. It has a high probability of 0.73.

d) Detection of locally stable regions: The last experiment concerning local probabilities deals with the detection of locally stable RNA regions in genomic sequence regions. We have chosen two RNA motifs from the Rfam database (histone 3' UTR stem-loop, Hepatitis C stem-loop), from which we have further chosen two sequences arbitrarily that contain these motifs. The sequences were given by the EMBL accession codes X83548 and AB057604. Surrounding nucleotides, i.e. 50 nucleotides before the motifs and the nu-

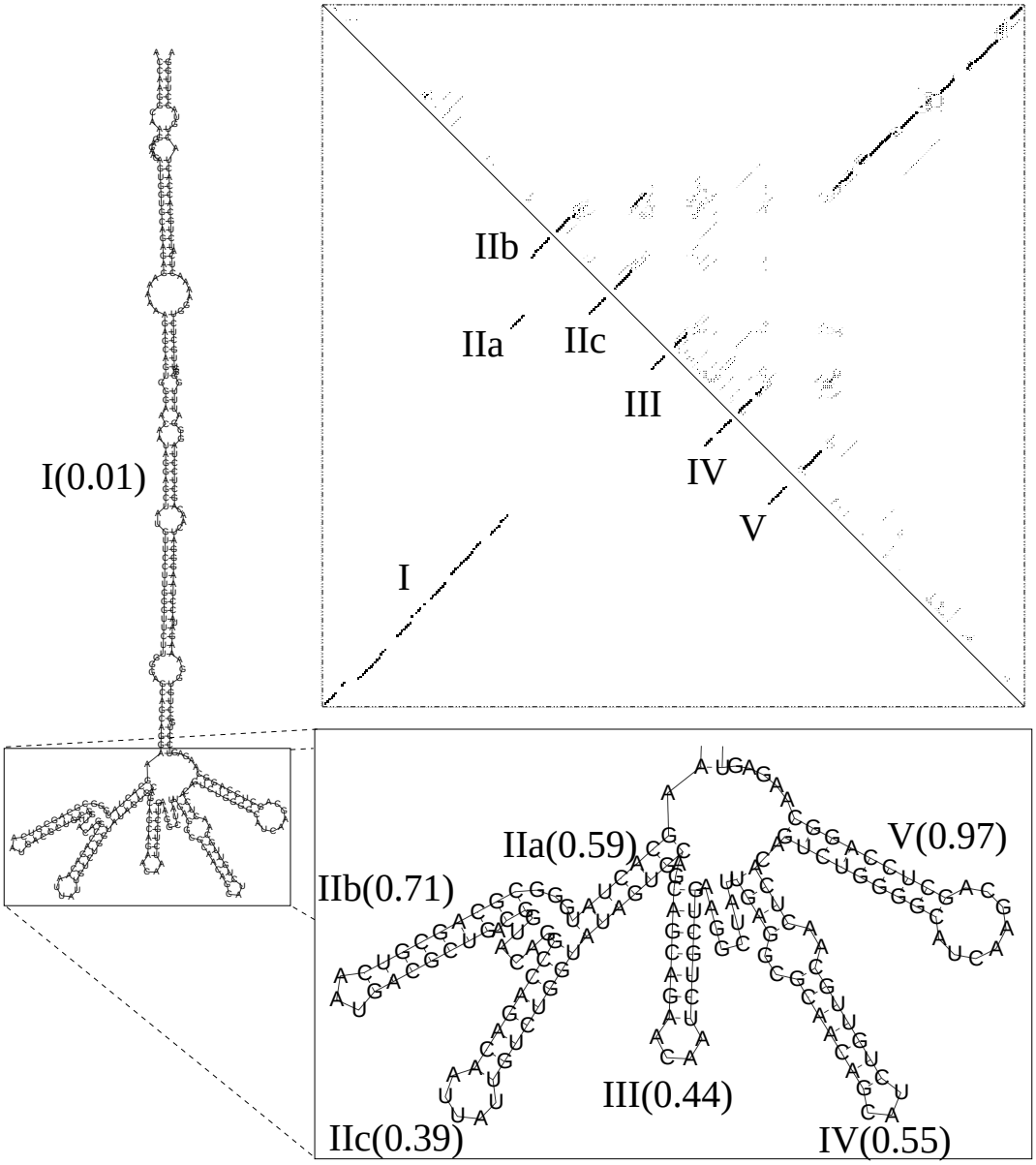


Figure 6.14: Rev response element (RRE), an RNA element encoded within the env region of Human immunodeficiency virus type 1 (HIV-1). On the left side, the mfe energy structure is depicted. The long stem I has occurrence probability 0.01, less stable than the remaining stems II-V. An enlargement of the middle part of the sequence is shown in the lower part of the figure. Occurrence probabilities are given in brackets. All stems are highly stable, whereas stem V occurs in almost all structures. The dot plot is given in the upper right part. Structural diversity is very low yielding highly stable regions.

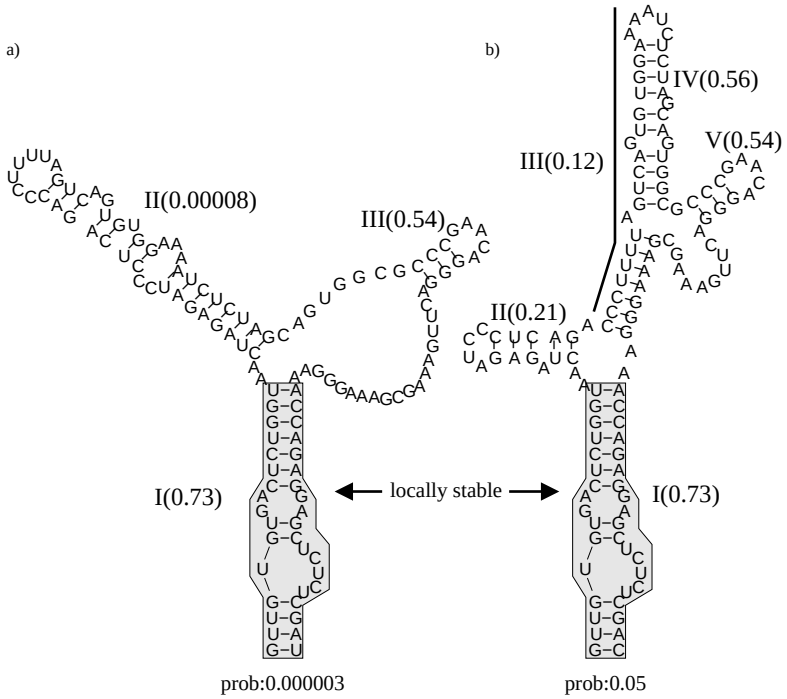
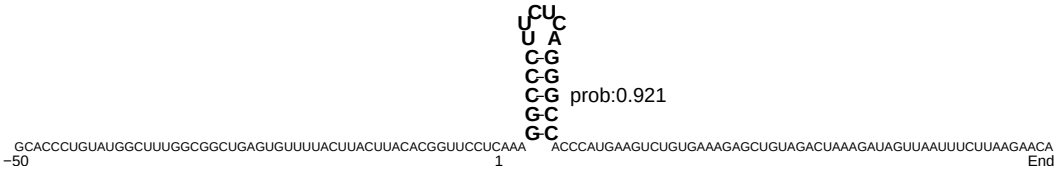


Figure 6.15: Two different structures for the PBS domain. a) Proposed secondary structure taken from Paillart et al. [2004]. It is dissected into the three stems I,II and III. Stem I and III are highly stable, whereas stem II has low energy. b) mfe structure with occurrence probability 0.05. Stem I and stem V can be found in both structures. Our stability program predicts stem I as the most stable region.

a) Histone 3' UTR stem-loop (X83548)



b) Hepatitis C stem-loop (AB057604)

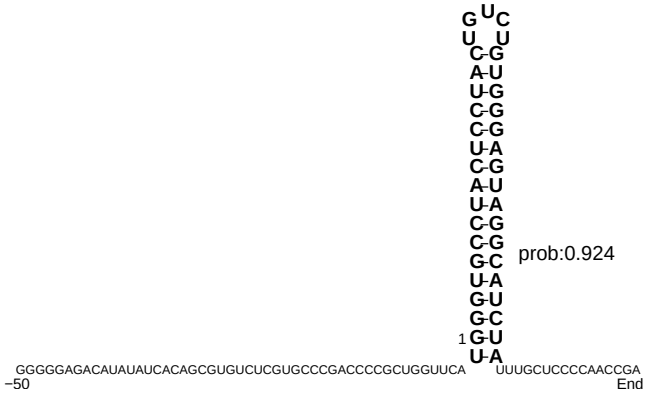


Figure 6.16: Detection of locally stable regions in genomic sequence regions. a) The histone 3'UTR stem-loop is found to be stable with a probability of 0.921 in a sequence including 50 nucleotides before the stem loop and all nucleotides to the end of the sequence (EMBL accession code: X83548). b) The hepatitis C stem-loop (EMBL: AB057604) was detected with a probability of 0.924. Again, the sequence with 50 nucleotides before the stem loop and all nucleotides to the end of the sequence was taken. Both RNA motifs can be found in the Rfam database as well.

cleotides to the end of the sequences were chosen to launch our prediction program (see Figure 6.16). In both cases, the motifs were identified correctly as they can be found in the Rfam database. The occurrence probabilities are both very high: The histone 3' UTR stem-loop is found with a probability of 0.921 in sequence with EMBL code X83548 and the hepatitis C stem-loop is found with a probability of 0.924 in sequence with EMBL code AB057604. Both motifs are shown in Figure 6.16.

Chapter 7

Conclusion

In this thesis, a characterization of RNAs concerning sequential and structural properties, on the one hand, and energetic properties on the other hand has been made.

The first characterization addresses mainly the combination of both the sequential and the structural properties of RNAs. It is therefore beyond the pure sequence considerations. This combination includes the description of RNAs as primary sequences and secondary structures, making it possible to handle biological problems computationally feasible. Global, pairwise RNA comparisons were proposed including different kinds of model considerations (section 3.1). They all have in common that base-pairs are considered as entities. Local analysis of RNAs has been made by finding exact or approximate patterns between two RNAs. The exact method is a fast approach to detect common patterns in time $O(nm)$ and space $O(nm)$. By allowing some inaccuracies in their sequential or structural composition, an algorithm to solve this problem has been proposed in time $O(n^2m^2max(n,m))$ and space (nm). Pairwise comparable RNA methods are a prerequisite to multiple comparable RNA methods using pairwise comparison strategies. Multiple RNA comparisons are intended to figure out common sequential and structural properties. Concerning this, a multiple alignment strategy has been developed and implemented in the tool *MARNA*. It has been accepted in publicity very well.

Concerning the thermodynamic considerations, algorithmic solutions were presented to the problem of discovering the most stable partial structure in a known as well as unknown structure. Four convincing examples were presented partially accompanied with dot plot figures. They show base-pair

probabilities as rectangular boxes. Base-pair probabilities were computed assuming that a particular base-pair may be involved in any loop. If a stem can be found in these plots as diagonal contiguous boxes, then we have to recall that there is no one-to-one correspondence between base-pair probabilities and stem probabilities. In most cases, there is a high correlation between the base-pair probability matrix and the local structures. This can be also derived from our examples, but this is not necessarily the case.

Lastly, we summarize and give some hints to the usage of *MARNA* as a numbered list. These hints are based on many experiences:

1. *MARNA* can be tested online via the webpage
   http://www.bioinf.uni-freiburg.de/Software/MARNA/index.html.
   MARNA is also available as a downloadable file (see webpage). The maximal sequence length of one RNA is restricted 500 bases. The maximal number of RNAs depends on the sequence lengths. The sum of all sequence lengths is restricted to 10000 bases.

2. *MARNA* offers mainly two choices to adjust your alignments:

   (a) *Parameter settings*: *MARNA* relies on the comparison of pairwise RNAs. These comparisons are accomplished by alignments with costs assigned to edit operations on bases and arcs. These costs can be set individually.

   (b) *Structure computation*: The alignment of RNAs take into account both the primary sequences and the secondary structures. The easiest case is when the secondary structures are known in advance, and the computation is reduced to find common sequential and structural properties. Otherwise, the structures have to be found. *MARNA* provides in addition to user-defined structures the assignment of different kinds of structures. These include the assignment of minimum free energy structures, shaped structures or an ensemble of low energy structures.

3. *Parameter settings*: Parameters can be set individually depending on weighting some edit operations more or less. A series of tests has brought three data sets to obtain alignments based on sequential or structural properties or on a mixture on both. These data sets are shown in Table 7.1.

edit operations	default	sequential	structural
base deletion	2.0	2.0	0.1
base mismatch	1.0	1.0	0.1
arc breaking	1.5	0.1	1.5
arc mismatch	1.8	0.1	1.8

Table 7.1: Data sets found out for weighting sequential or structural properties or on a mixture of both (default values). The values correspond to costs which can be set in the *MARNA* system.

4. Parameters settings influence the resulting alignments. Choose the default parameter settings first. It has been confirmed that this data set recognizes conserved sequential and structural properties very well.

5. Beyond the parameter settings, the assignment of different structures to the sequences are quite important as well. The easiest case is when user-defined structures are given as input.

6. Structure Choice: Here are some hints to choose the right structure assignments if no structures are given to the sequences.

   (a) If the RNAs are sequentially related and have nearly the same length then choose the minimum free energy structures.

   (b) The shaped structures are suited to cover a lot of diverse structural conformations for each single sequence. Choose shape structures, if no clear consensus structure is observable at first glance.

   (c) The ensemble set of low energy conformations is best chosen if you guess that these RNA sequences resemble structurally in some way. An ensemble consists of multiple structures. This ensemble contains similar structures if almost all suboptimal structures are similar.

7. The running time of *MARNA* crucially depends on the structure choices. Suppose n RNAs of nearly the same length without structure specifications are given. If the mfe structures are chosen that are assigned to the sequences then the multiple alignment and the consensus structure computation can be done in reasonable time. Suppose you choose en

ensemble of three suboptimal structures to each RNA, then the computation time is ninefold because for each pair of RNAs nine pairwise sequence structure comparisons have to be made.

Bibliography

S. Altman. Biosynthesis of transfer RNA in Escherichia coli. *Cell*, 4(1):21–9, 1975.

S. Altman, A. L. Bothwell, and B. C. Stark. Processing of E. coli tRNA Tyr precursor RNA in vitro. *Brookhaven Symp Biol*, 0(26):12–25, 1975.

S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–10, 1990.

R. Backofen and S. Siebert. Fast detection of common sequence structure patterns in RNAs. *Journal of Discrete Algorithms*, 2006. in print.

R. Backofen and S. Siebert. Fast detection of common sequence structure patterns in RNAs. In *Symposium on String Processing and Information Retrieval 2004 (SPIRE 2004)*, 2004.

R. Backofen and S. Will. Local sequence-structure motifs in RNA. *Journal of Bioinformatics and Computational Biology (JBCB)*, 2(4):681–698, 2004. URL `http://www.worldscinet.com/jbcb/02/0204/S0219720004000818.html`.

V. Bafna, S. Muthukrishnan, and R. Ravi. Computing similarity between RNA strings. In *Proc. 6th Symp. Combinatorical Pattern Matching*, pages –16, 1995.

A. Bairoch and R. Apweiler. The swiss-prot protein sequence database and its supplement trembl in 2000. *Nucleic Acids Res*, 28(1):45–8, 2000.

A. L. Bothwell, B. C. Stark, and S. Altman. Ribonuclease P substrate specificity: cleavage of a bacteriophage phi80-induced RNA. *Proc. Natl. Acad. Sci. USA*, 73(6):1912–6, 1976.

J. W. Brown. The ribonuclease P database. *Nucleic Acids Research*, 27(1): 314, 1999a.

M. P. Brown. *RNA Modeling Using Stochastic Context-Free Grammars*. PhD thesis, University of California, Santa Cruz, 1999b.

D. K. Chiu and T. Kolodziejczak. Inferring consensus structure from nucleic acid sequences. *Comput Appl Biosci*, 7(3):347–52, 1991.

F. Corpet and B. Michot. RNAlign program: alignment of RNA sequences using both primary and secondary structures. *Comput Appl Biosci*, 10(4): 389–99, 1994.

F. Crick. The origin of the genetic code. *J.Mol.Bio*, 38:367–379, 1968.

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, supplement 3, pages 345–352. National Biomedical Research Foundation, Washington, DC, 1978.

Y. Ding and C. E. Lawrence. Statistical prediction of single-stranded regions in RNA secondary structure and application to predicting effective antisense target sites and beyond. *Nucleic Acids Research*, 29(5):1034–46, 2001.

Y. Ding and C. E. Lawrence. A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Research*, 31(24):7280–301, 2003.

L. DJ, A. SF, and K. JD. A tool for multiple sequence alignment. *Proc Natl Acad Sci U S A*, 86(12):4412–5, 1989.

E. A. Doherty and J. A. Doudna. Ribozyme structures and mechanisms. *Annual review of biophysics and biomolecular structure*, 30:457–75, 2001.

J. A. Doudna and T. R. Cech. The chemical repertoire of natural ribozymes. *Nature*, 418(6894):222–8, 2002.

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

D. Fagegaltier, A. Lescure, R. Walczak, P. Carbon, and A. Krol. Structural analysis of new local features in SECIS RNA hairpins. *Nucleic Acids Res*, 28(14):2679–89, 2000.

D. F. Feng and R. F. Doolittle. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J Mol Evol*, 25(4):351–60, 1987.

R. Giegerich, B. Voss, and M. Rehmsmeier. Abstract shapes of RNA. *Nucleic Acids Research*, 32(16):4843–51, 2004.

J. Gorodkin, S. L. Stricklin, and G. D. Stormo. Discovering common stem-loop motifs in unaligned RNA sequences. *Nucleic Acids Research*, 29(10): 2135–44, 2001.

O. Gotoh. An improved algorithm for matching biological sequences. *Journal of Molecular Biology*, 162:705–708, 1982.

O. Gotoh. Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *Journal of Molecular Biology*, 264(4):823–38, 1996.

S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S. R. Eddy. Rfam: an RNA family database. *Nucleic Acids Research*, 31(1):439–41, 2003.

R. R. Gutell and C. R. Woese. Higher order structural elements in ribosomal RNAs: pseudo-knots and the use of noncanonical pairs. *Proc. Natl. Acad. Sci. USA*, 87(2):663–7, 1990.

R. R. Gutell, A. Power, G. Z. Hertz, E. J. Putz, and G. D. Stormo. Identifying constraints on the higher-order structure of RNA: continued development and application of comparative sequence analysis methods. *Nucleic Acids Research*, 20(21):5785–95, 1992.

S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, 89(22):10915–9, 1992.

M. Höchsmann, T. Töller, R. Giegerich, and S. Kurtz. Local similarity in RNA secondary structures. In *Proceedings of Computational Systems Bioinformatics (CSB 2003)*, 2003.

I. L. Hofacker. Vienna RNA secondary structure server. *Nucleic Acids Research*, 31(13):3429–31, 2003.

I. L. Hofacker and P. F. Stadler. The partition function variant of sankoff's algorithm. In *Computational Science - ICCS 2004, Part IV*, Lecture Notes in Computer Science LNCS 3039, pages 728–735, Heidelberg, June 2004. Springer Verlag.

I. L. Hofacker, W. Fontana, P. F. Stadler, S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures. *Monatshefte Chemie*, 125:167–188, 1994.

I. L. Hofacker, M. Fekete, and P. F. Stadler. Secondary structure prediction for aligned RNA sequences. *Journal of Molecular Biology*, 319(5):1059–66, 2002.

I. L. Hofacker, S. H. Bernhart, and P. F. Stadler. Alignment of RNA base pairing probability matrices. *Bioinformatics*, 2004a.

I. L. Hofacker, B. Priwitzer, and P. F. Stadler. Prediction of locally stable RNA secondary structures for genome-wide surveys. *Bioinformatics*, 20 (2):186–190, 2004b.

T. Jiang, J. Wang, and K. Zhang. Alignment of trees - an alternative to tree edit. *Theoretical Computer Science*, 143(1):137–148, 1995.

T. Jiang, G. Lin, B. Ma, and K. Zhang. A general edit distance between RNA structures. *Journal of Computational Biology*, 9(2):371–88, 2002.

J. Kececioglu and W. Zhang. Aligning alignments. In M. Farach-Colton, editor, *Proceedings of the 9th Annual Symposium on Combinatorial Pattern Matching*, pages 189–208, Piscataway, NJ, 1998. Springer-Verlag, Berlin.

J. D. Kececioglu, H.-P. Lenhof, K. Mehlhorn, P. Mutzel, K. Reinert, and M. Vingron. A polyhedral approach to sequence alignment problems. *Discrete Applied Mathematics*, 104:143–186, 2000.

R. J. Klein and S. R. Eddy. RSEARCH: finding homologs of single structured RNA sequences. *BMC Bioinformatics*, 4(1):44, 2003.

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

R. Kole and S. Altman. Reconstitution of RNase P activity from inactive RNA and protein. *Proc. Natl. Acad. Sci. USA*, 76(8):3795–9, 1979.

G. V. Kryukov, V. M. Kryukov, and V. N. Gladyshev. New mammalian selenocysteine-containing proteins identified with an algorithm that searches for selenocysteine insertion sequence elements. *Journal of Biological Chemistry*, 274(48):33888–97, 1999.

G. V. Kryukov, S. Castellano, S. V. Novoselov, A. V. Lobanov, O. Zehtab, R. Guigo, and V. N. Gladyshev. Characterization of mammalian selenoproteomes. *Science*, 300(5624):1439–43, 2003.

A. Lambert, A. Lescure, and D. Gautheret. A survey of metazoan selenocysteine insertion sequences. *Biochimie*, 84(9):953–9, 2002.

H. P. Lenhof, K. Reinert, and M. Vingron. A polyhedral approach to RNA sequence structure alignment. *Journal of Computational Biology*, 5(3): 517–30, 1998.

A. Lescure, D. Gautheret, P. Carbon, and A. Krol. Novel selenoproteins identified in silico and in vivo by using a conserved RNA structural motif. *Journal of Biological Chemistry*, 274(53):38147–54, 1999.

S. C. Low and M. J. Berry. Knowing when not to stop: selenocysteine incorporation in eukaryotes. *Trends in Biochemical Sciences*, 21(6):203–208, 1996.

R. Luck, S. Graf, and G. Steger. Construct: a tool for thermodynamic controlled prediction of conserved secondary structure. *Nucleic Acids Research*, 27(21):4208–17, 1999.

R. B. Lyngso, M. Zuker, and C. N. Pedersen. An improved algorithm for RNA secondary structure prediction. Technical report, Department of Computer Science, University of Aarhus., 1999.

B. Ma, Z. Wang, and K. Zhang. Alignment between two multiple alignments. In *Combinatorial Pattern Matching, 14th Annual Symposium, CPM 2003, Morelia, Michocán, Mexico, June 25-27, 2003, Proceedings*, 2003.

D. Mathews, J. Sabina, M. Zuker, and D. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J Mol Biol*, 288(5):911–40, 1999.

J. S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–19, 1990.

S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–53, 1970.

C. Notredame and D. G. Higgins. SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Research*, 24(8):1515–24, 1996.

C. Notredame, D. G. Higgins, and J. Heringa. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, 302(1):205–17, 2000.

R. Nussinov, G. Pieczenik, J. R. Griggs, and D. J. Kleitman. Algorithms for loop matchings. *SIAM Journal on Applied Mathematics*, 35(1):68–82, July 1978. ISSN 0036-1399 (print), 1095-712X (electronic).

L. Orgel. Evolution of the genetic apparatus. *J.Mol.Bio*, 38:381–393, 1968.

J.-C. Paillart, M. Dettenhofer, X.-F. Yu, C. Ehresmann, B. Ehresmann, and R. Marquet. First snapshots of the HIV-1 RNA structure in infected cells and in virions. *Journal of Biological Chemistry*, 279(46):48397–403, 2004.

W. R. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, 85(8):2444–8, 1988.

S. Perrey, J. Stoye, V. Moulton, and A. Dress. On simultaneous versus iterative multiple sequence alignment. Technical report, Universitt Bielefeld, Forschungsschwerpunkt Mathematisierung - Strukturbildungsprozesse, 1997. URL `citeseer.nj.nec.com/perrey97simultaneous.html`. Materialien/Preprints 111.

M. A. Rosenblad, J. Gorodkin, B. Knudsen, C. Zwieb, and T. Samuelsson. SRPDB: Signal Recognition Particle Database. *Nucleic Acids Research*, 31(1):363–4, 2003.

M. Rother, R. Wilting, S. Commans, and A. Böck. Identification and characterisation of the selenocysteine-specific translation factor SelB from the archaeon Methanococcus jannaschii. *Journal of Molecular Biology*, 299(2): 351–8, 2000.

N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol*, 4(4):406–25, 1987.

D. Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J. Appl. Math.*, 45(5):810–825, 1985.

D. Sankoff and M. Zuker. Rna secondary structures and their prediction. *Bull. Math. Biol.*, 46:591–621, 1984.

S. Siebert. Multiple local sequence structure alignment of rna sequences. In *European Conference on Computational Biology 2002*, Saarbrcken, October 2002. Poster.

S. Siebert and R. Backofen. MARNA: multiple alignment and consensus structure prediction of RNAs based on sequence structure comparisons. *Bioinformatics*, 21(16):3352–9, 2005a.

S. Siebert and R. Backofen. *Methods for Multiple Alignment and Consensus Structure Prediction of RNAs implemented in MARNA*, chapter Comparative Genomics. The Humana Press Inc., 2006a.

S. Siebert and R. Backofen. A new distance measure of rna ensembles and its application to phylogenetic tree construction. In G. B. Fogel, editor, *IEEE 2005 Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 150–157, La Jolla, CA, USA, November 2005b. IEEE.

S. Siebert and R. Backofen. MARNA: A server for multiple alignment of RNAs. In *GCB 2003*, October 2003a.

S. Siebert and R. Backofen. Screening the human utr database for stabilized rna secondary structures. In *ISMB 2006*, Fortaleza(Brazil), August 2006b. Poster.

S. Siebert and R. Backofen. A dynamic programming approach for finding common patterns in RNAs. *Journal of Computational Biology*, 2006c. submitted.

S. Siebert and R. Backofen. Multiple local sequence structure alignment of rna sequences. In *RECOMB 2003*, Berlin, April 2003b. Poster.

T. Smith and M. Waterman. Comparison of biosequences. *Adv. appl. Math.*, 2:482–489, 1981a.

T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981b.

B. C. Stark, R. Kole, E. J. Bowman, and S. Altman. Ribonuclease P: an enzyme with an essential RNA component. *Proc. Natl. Acad. Sci. USA*, 75(8):3717–21, 1978.

K.-C. Tai. The tree-to-tree correction problem. *In Journal of the ACM*, 26 (3):422–433, 1979.

J. D. Thompson, D. G. Higgins, and T. J. 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(22):4673–80, 1994.

J. D. Thompson, F. Plewniak, and O. Poch. A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Res*, 27(13):2682–90, 1999.

R. Walczak, E. Westhof, P. Carbon, and A. Krol. A novel RNA structural motif in the selenocysteine insertion element of eukaryotic selenoprotein mRNAs. *RNA*, 2(4):367–79, 1996.

Z. Wang and K. Zhang. Alignment between two RNA structures. In J. Sgall, A. Pultr, and P. Kolman, editors, *Proc. of the 26th International Symposium on Mathematical Foundations of Computer Science (MFCS01)*, pages 690–702, 2001.

Z. Wang and K. Zhang. Multiple RNA structure alignment. In *3rd International IEEE Computer Society Computational Systems Bioinformatics Conference (CSB 2004)*, pages 246–254. IEEE Computer Society, 2004.

R. Wilting, S. Schorling, B. C. Persson, and A. Bck. Selenoprotein synthesis in archaea: Identification of an mRNA element of Methanococcus jannaschii probably directing selenocysteine insertion. *Journal of Molecular Biology*, 266(4):637–41, 1997.

C. Woese. *The Genetic Code*. New York: Harper and Row, 1967.

A. J. Zaug and T. R. Cech. In vitro splicing of the ribosomal RNA precursor in nuclei of Tetrahymena. *Cell*, 19(2):331–8, 1980.

K. Zhang. Computing similarity between RNA secondary structures. In *Proceedings of the IEEE International Joint Symposia on Intelligence and Systems*, pages 126–132, 1998.

K. Zhang and D. Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262, 1989.

M. Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Research*, 31(13):3406–15, 2003.

M. Zuker. Prediction of RNA secondary structure by energy minimization. *Methods in Molecular Biology*, 25:267–94, 1994.

M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research*, 9(1):133–48, 1981.

C. Zwieb, J. Gorodkin, B. Knudsen, J. Burks, and J. Wower. tmRDB (tmRNA database). *Nucleic Acids Research*, 31(1):446–7, 2003.