RNA Secondary Structure Design under Simple and Complex Constraints

Dissertation

zur Erlangung des akademischen Grades

doctor rerum naturalium (Dr. rer. nat.)

vorgelegt dem Rat
der Fakultät für Angewandte Wissenschaften
der Albert-Ludwigs-Universität Freiburg

von Diplom-Wirtschaftsmathematikerin

Anke Busch

Dekan:

Prof. Dr. Bernhard Nebel

Gutachter:

Prof. Dr. Rolf Backofen

Prof. Dr. Peter F. Stadler

Datum der Promotion: 03. Juni 2008

RNA Secondary Structure Design under Simple and Complex Constraints

Anke Busch

January 2008

Abstract

The structure of RNA molecules is often crucial for their function. Therefore, designing RNA sequences that fold into a given structure with special features is of interest for biologists in several fields.

In this thesis, we introduce two approaches dealing with RNA sequence design. First, we develop a fast and successful new approach to the inverse RNA folding problem, called INFO-RNA, incorporating constraints on the sequence, while afterwards, our interests focus on a more complex design of RNA sequences taking into account the translated amino acid sequences. We develop an algorithm, called SECISDesign, that solves this complex task successfully.

In the first part of this thesis, we focus on the search for an RNA sequence $S = S_1...S_n$ that folds into a given secondary structure T and fulfills given constraints $C = C_1...C_n$ on the primary sequence. Our new approach INFO-RNA consists of two parts; a dynamic programming method for good initial sequences and a following improved stochastic local search that uses an effective neighbor selection method. During the initialization, we design a sequence that among all sequences adopts the given structure with the lowest possible energy. There is no other sequence that has lower energy when folding into this structure. Nevertheless, the sequence is not guaranteed to fold into T since it can have even less energy when folding into another structure. Therefore, the resulting sequence is processed further using a stochastic local search. For the selection of neighbors during this search, we use a kind of look-ahead of one selection step applying an additional energy-based criterion. Afterwards, the pre-ordered neighbors are tested using the actual optimization criterion of minimizing the structure distance between the target structure and the structure of the considered neighbor with minimal free energy or maximizing the probability of folding into T.

We compare INFO-RNA to the two existing tools RNAinverse and RNA-SSD using artificial as well as biological test sets. Running INFO-RNA, we perform better than RNAinverse and in most cases, we obtain better results than RNA-SSD, the probably best inverse RNA folding tool on the market.

The second part of this thesis concentrates on a more complex design of RNA sequences. Here, not only the secondary structure of the RNA sequence is taken into account but also the amino acid sequence, which is encoded by the designed RNA. Thus, this part of the thesis can be applied to the design of coding parts of mRNA sequences. The most prominent example is the application to selenoproteins. These are proteins containing the 21st amino acid selenocysteine, which is encoded by the STOP-codon UGA. For its insertion it requires a specific mRNA sequence downstream the UGA-codon that forms a hairpin-like structure (called selenocysteine insertion sequence (SECIS)) and codes for the amino acids following the selenocysteine in the protein.
Selenoproteins have gained much interest recently since they are important for human health. In contrast, very little is known about them. One reason for this is that one is not able to produce enough amount of selenoproteins by using recombinant expression in a system like *E.coli* straightforwardly since the selenocysteine insertion mechanisms are different between *E.coli* and eukaryotes. Thus, one has to redesign the human/mammalian selenoprotein for the expression in *E.coli*. In this thesis, we introduce an polynomial-time algorithm, called SECISDesign, for solving the computational problem involved in this design and present results for known selenoproteins.

In the third part of this thesis, we introduce two web services, which make it possible to use our two algorithms online. They introduce the programs INFO-RNA and SECISDesign to a wide community of scientists and allow them to design RNA sequences in an automatic manner. They allow to use INFO-RNA and SECISDesign without compiling their source codes and even without having an executable version of them on the local computer.

To conclude, both, INFO-RNA and SECISDesign, are fast and successful tools for the design of RNA sequences dealing with simple and complex constraints, respectively. They outperform existing tools and methods for most structures.

Zusammenfassung

Die Struktur von RNA Molekülen ist oft entscheidend für ihre Funktion. Deshalb ist das Design von RNA Sequenzen, die in eine gegebene Struktur falten, von großem Interesse in der derzeitigen Forschung.

In dieser Arbeit stellen wir zwei neue Ansätze vor, die sich mit dem Design von RNA Sequenzen beschäftigen. Im ersten Teil entwickeln wir einen schnellen und erfolgreichen Ansatz (INFO-RNA) zur Lösung des inversen RNA Faltungsproblems. Im zweiten Teil geht es um ein weitaus komplexeres Designproblem. Hier stellen wir den Ansatz SECISDesign vor, der sich mit dem Design von kodierenden RNA Sequenzen beschäftigt und deshalb zusätzlich die von der designten mRNA kodierte Aminosäuresequenz berücksichtigt.

Im ersten Teil dieser Dissertation geht es um das inverse RNA Faltungsproblem mit einigen Einschränkungen der möglichen Sequenz. Mit anderen Worten, es geht um die Suche nach einer RNA Sequenz $S = S_1...S_n$, die sich in eine gegebene Sekundärstruktur T faltet und zusätzlich gegebene Bedingungen $C = C_1...C_n$ an die RNA Sequenz erfüllt. Unser neuer Ansatz, INFO-RNA, besteht aus zwei Teilen: einem dynamischen Programmierverfahren zur Erzeugung einer initialen Sequenz und einer anschließenden erweiterten stochastischen lokalen Suche, die eine neue effektive Methode zur Kandidatenauswahl benutzt. Durch die Initialisierung finden wir eine Sequenz, die die gegebene Struktur mit der geringsten möglichen freien Energie annimmt. Es gibt keine andere Sequenz, die eine geringere freie Energie hat, wenn sie in T faltet. Trotzdem kann nicht garantiert werden, dass sich unsere designte Sequenz S in die gewünschte Struktur T faltet, da S selbst in eine andere Struktur mit geringerer freier Energie falten könnte. Deshalb wird die initial-gefundene Sequenz durch eine anschließende lokale Suche bezüglich ihrer Faltungseigenschaft weiter verbessert. Während dieser Suche verwenden wir ein zusätzliches Energiekriterium, dass uns erlaubt, einen Schritt vorauszublicken. Nachdem alle Sequenzkandidaten durch dieses Kriterium in eine Reihenfolge (entsprechend ihrer Güte) gebracht wurden, werden sie mit dem eigentlichen Suchkriterium geprüft. Hierbei wird versucht, die Distanz zwischen der Zielstruktur T und der Struktur mit minimaler freier Energie der designten Sequenz zu minimieren oder die Faltungswahrscheinlichkeit für T zu maximieren.

Wir vergleichen INFO-RNA mit den beiden existierenden Programmen zur inversen RNA Faltung, RNAinverse und RNA-SSD. Dazu verwenden wir sowohl künstlich generierte also auch biologisch real existierende Datensätze. Es zeigt sich, dass INFO-RNA schneller bessere Ergebnisse liefert als die beiden Vergleichsprogramme.

Der zweite Teil der Dissertation beschäftigt sich mit einem komplexeren Design von RNA Sequenzen. Es wird nicht nur die gewünschte Sekundärstruktur betrachtet sondern auch die Aminosäuresequenz, die durch die designte RNA Sequenz kodiert wird. Somit kann dieser Teil der Arbeit zum Design von kodierenden Teilen der mRNA benutzt werden. Das bekannteste Beispiel ist die Anwendung des Programms auf Selenoproteine. Diese sind Proteine, die die 21ste Aminosäure Selenocystein enthalten, die durch das sonstige Stopcodon UGA kodiert wird. Um Selenocystein einzubauen ist eine spezielle mRNA Sequenz strangabwärts des UGA-Codons notwendig, die eine haarnadel-ähnliche Struktur ausbildet. Sie wird "selenocysteine insertion sequence" (SECIS) genannt und kodiert außerdem für die Aminosäuren, die dem Selenocystein folgen.
Während der vergangenen Jahre sind Selenoproteine immer mehr ins Interesse der Forschung gerückt. Trotzdem ist relativ wenig über sie bekannt. Das liegt daran, dass es nur schwer möglich ist, große Mengen von ihnen in einem rekombinanten Expressionssystem wie *E.coli* herzustellen, da der Einbau von Selenocystein in *E.coli* anders funktioniert als in Eukaryoten. Deshalb muss die RNA eukaryotischer Selenoproteine für die Expression in *E.coli* umdesignt werden, so dass sie das SECIS-element an der für *E.coli* notwendigen Position haben. In dieser Arbeit stellen wir den polynomiellen Algorithmus SECISDesign vor, der dieses Designproblem löst und präsentieren außerdem Ergebnisse für bekannte Selenoproteine des Menschen und der Maus.

Im dritten Teil dieser Arbeit stellen wir zwei Webserver vor, durch die es möglich ist, unsere Programme INFO-RNA und SECISDesign online zu benutzen und sie somit einer breiten Masse von Wissenschaftlern zugänglich zu machen.

Abschließend bleibt zu sagen, dass INFO-RNA und SECISDesign zwei schnelle und erfolgreiche Programme zum Design von RNA Sequenzen sind, die besser und schneller arbeiten als andere existierende Verfahren.

Danksagung

Mein erster Dank gilt meinem Doktorvater Rolf Backofen. Ihm danke ich für die Betreuung dieser Arbeit, viele interessante Diskussionen, für die gute und erfolgreiche Zusammenarbeit aus der einige gemeinsame Publikationen entstanden sind, sowie für seinen Enthusiasmus und die Inspiration zu neuen Ideen.

Bedanken möchte ich mich bei Peter Stadler für das Interesse an meiner Arbeit und die Bereitschaft, diese Dissertation zu begutachten.

Desweiteren möchte ich mich bei meinen Kollegen und ehemaligen Kollegen am Lehrstuhl für Bioinformatik zunächst in Jena und später in Freiburg Michael Beckstette, Steffen Heyne, Michael Hiller, Martin Mann, Rainer Pudimat, Andreas Richter, Sven Siebert und Sebastian Will für viele interessante und hilfreiche Diskussionen sowie für all die gemeinsamen Unternehmungen bedanken. Außerdem gilt mein Dank Monika Degen-Hellmuth für die Hilfe beim Überwinden aller bürokratischen Hürden. Für das Korrekturlesen dieser Arbeit bedanke ich mich bei Michael Hiller und Martin Mann. Besonders bedanken möchte ich mich bei meinem Zimmerkollegen Michael Hiller für seine Unterstützung, viele hilfreiche Tipps, interessante Diskussionen und nicht zuletzt für das Ertragen meiner "kommunikativen" Phasen und die häufig etwas höhere Raumtemperatur.

Außerdem bedanke ich mich bei meiner Mutter Monika für ihre immerwährende Unterstützung, ihr allzeit offenes Ohr und ihre Liebe, ohne das alles diese Arbeit nicht möglich geworden wäre. Last but not least möchte ich mich bei meinem Freund Martin bedanken, der mich nun schon seit vielen Jahren liebevoll begleitet und unterstützt, mich auch in schwierigen Situationen wieder aufbaut und mir neuen Mut gibt.

List of Own Publications

[1] Sebastian Will, Anke Busch, and Rolf Backofen. Efficient sequence alignment with side-constraints by cluster tree elimination. *Constraints Journal*, 2008, to appear.

[2] Anke Busch and Rolf Backofen. INFO-RNA - a server for fast inverse RNA folding satisfying sequence constraints. *Nucleic Acids Research*, 35(Web Server Issue), W310-3, 2007.

[3] Anke Busch and Rolf Backofen. INFO-RNA - a fast approach to inverse RNA folding. *Bioinformatics*, 22(15), 1823-31, 2006.

[4] Michael Hiller, Rainer Pudimat, Anke Busch, and Rolf Backofen. Using RNA secondary structures to guide sequence motif finding towards single-stranded regions. *Nucleic Acids Research*, 34(17), e117, 2006.

[5] Anke Busch, Sebastian Will, and Rolf Backofen. SECISDesign: a server to design SECIS-elements within the coding sequence. *Bioinformatics*, 21(15), 3312-3, 2005.

[6] Sebastian Will, Anke Busch, and Rolf Backofen. Efficient constraint-based sequence alignment by cluster tree elimination. *Proceedings of the Workshop on Constraint Based Methods in Bioinformatics (WCB05)*, 66-74, 2005.

[7] Rolf Backofen and Anke Busch. Computational design of new and recombinant selenoproteins. *Proceedings of the 15th Annual Symposium on Combinatorial Pattern Matching (CPM04)*, 2004.

[8] Michael Hiller, Rolf Backofen, Stephan Heymann, Anke Busch, Timo Mika Glaesser, and Johann-Christoph Freytag. Efficient prediction of alternative splice forms using protein domain homology. *In Silico Biology*, 4(2), 0017, 2004.

Contents

Chapter 1

Introduction and Fundamental Concepts

In 1968, Francis H.C. Crick and Leslie E. Orgel published two overlapping, highly speculative papers about the origin of life: one concentrating on the evolution of protein synthesis [Cri68] and the other on the origin of molecular replication [Org68]. Contrary to the predominant speculation at that time, they proposed a solution of the "chicken and egg" problem, - Which came first, the information or the function, the nucleic acid or the protein? - if nucleic acids served as catalysts for their own replication. About 15 years later, Cech [KGZ+82] and Altman [GTGM+83], independently, discovered ribozymes, which are specific catalysts made of RNA. They behave as enzymes and catalyze their own assembly. In 1986, Walter Gilbert stated that it seems possible that the informational properties of RNAs and the enzymatic activities of proteins may be combined in RNA. He coined the phrase "The RNA World" [Gil86]. Some years later, Noller and colleagues suggested that ribosomal RNA (rRNA) participates in the catalysis of the peptide bond-forming step during protein synthesis [NHZ92].

25 years after their speculative papers, Crick and Orgel summed up about their past hypotheses. They found a lot of them approved and others disapproved. Arguing on the origin of life, they admit that they neither searched for nor encouraged others to search for relics of the RNA world in contemporary organisms [OC93]. In 1995, Charles Wilson and Jack W. Szostack announced that they had manufactured ribozymes, which promoted the formation of peptide bonds [WS95]. They suggested that RNA may be capable of a broad rage of catalytic activities.

These and several other findings show that RNA has a lot more functions than just holding and transporting genetic information. During the last years, several new classes of RNAs were found, which are not translated into proteins. All of them are summarized in the group of *non-coding RNAs* (ncRNAs). But this does not mean

that they do not contain information or have no function [MM06]. Non-coding RNAs are ubiquitous in the cell and important for many processes. In particular, they act as a means of gene regulation, both in *cis* and *trans* and thus, they are employed in numerous mechanisms controlling expression of genes [SEB07]. They are involved in translation (transfer RNA (tRNA), ribosomal RNA (rRNA)), splicing (small nuclear RNA(snRNA)), processing of other RNAs (small nucleolar RNA (snoRNA), nuclear ribonuclease P (RNAseP)), and in regulatory processes (micro RNA (miRNA), small interfering RNA (siRNA)) [HBB02].
But not only non-coding RNAs can have regulatory functions. Parts of mRNAs can adopt structures that regulate their own translation, e.g. the hairpin-like structure necessary in the mRNA coding for a protein that includes *selenocysteine* [HB98, LRGEK98] and the iron responsive element (IRE), which is essential for the expression of proteins that are involved in the iron metabolism [ABK$^+$97, HK96].
All these findings indicate that the function of RNA often depends on both; the sequence and structure. In many cases, the structure is even more important than the sequence. This characteristic is used in this thesis. We investigate the design of RNA sequences adopting a special structure responsible for the function of the molecule. In the remaining part of this chapter, we explain the basic preliminaries and concepts of RNA and its structure. Section 2 deals with a new method for the design of RNA sequences that fold into a given structure (INFO-RNA). While this approach is mainly used for the design of non-coding RNAs since no or only simple conditions on the sequence are incorporated, in Chapter 3 we introduce a new more complex approach for the design of protein-coding RNA sequences that additionally have to form a given secondary structure. Here, complex and often conflicting conditions are involved. They constrain the protein sequence that forms when translating the designed mRNA as well as the secondary structure adopted by the RNA. Furthermore, the strict specification of the RNA structure can be eased by the user. The usefulness of the approach is demonstrated by the design of ***se**leno**c**ysteine **i**nsertion **s**equence elements* (SECIS-elements), which are small hairpin-like structures within the coding region of the mRNA and needed for the incorporation of selenocysteine into proteins. Since the design of SECIS-elements is the main field of application, the introduced method is called SECISDesign. For both RNA sequence designers, we are offering a web service. These are described in Section 4 of this thesis. Finally, we summarize our research in Chapter 5.

1.1 Different Views at RNA and its Structure

Ribonucleic acid (RNA) is a nucleic acid polymer consisting of nucleotide monomers. Each nucleotide consists of phosphate, a sugar, and one of the four different bases:

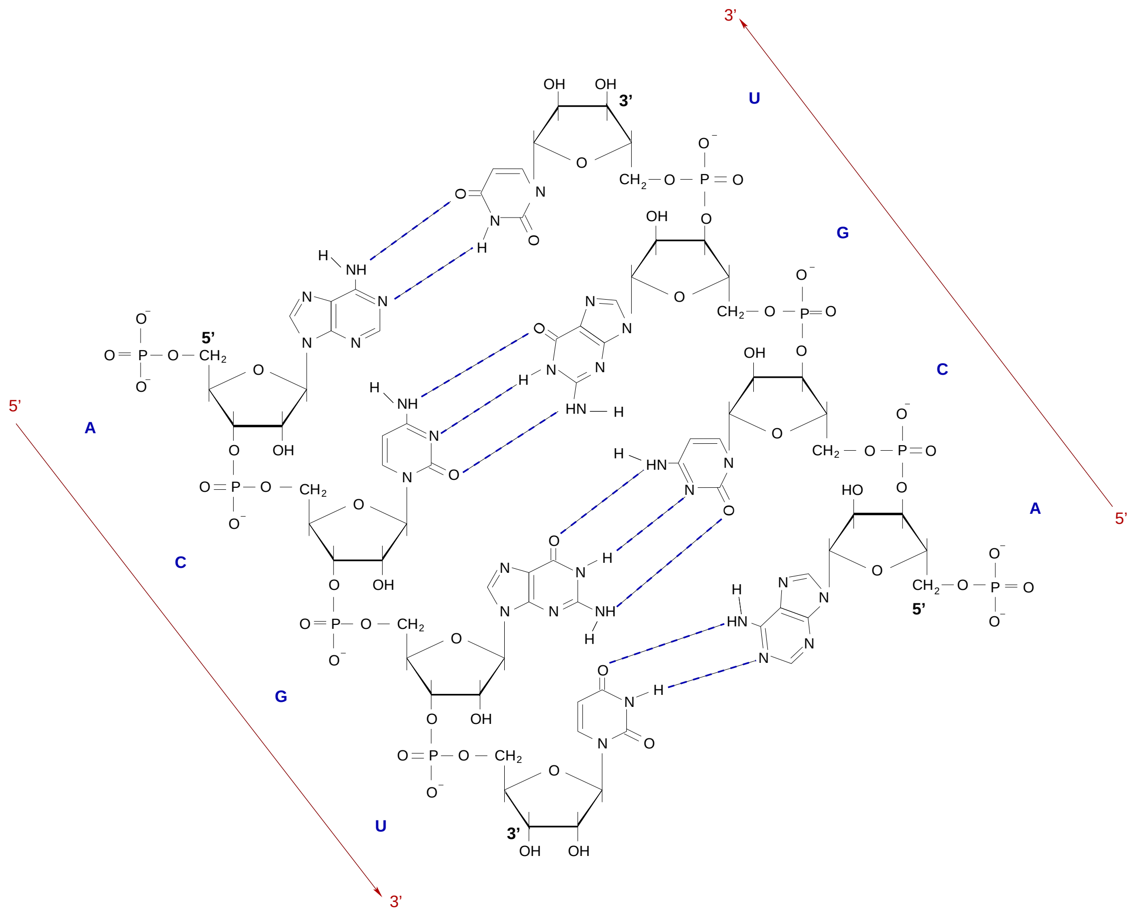


Figure 1.1: Exemplary chemical structure of an RNA double strand.

adenine (A), *cytosine* (C), *guanine* (G), and *uracil* (U). The sugars are joined together by the phosphate groups that form phosphodiester bonds between the third and fifth carbon atoms of adjacent sugar rings. By these asymmetry, a direction is given to the RNA strand, which is always specified in 5' to 3' direction (see Figure 1.1).

In contrast to DNA (deoxyribonucleic acid), which is composed of two strands and organized in a double helix, RNA consists of just a single strand. But equally to DNA, RNA can form hydrogen bonds between complementary bases according to Watson and Crick [WC53] (A and U, C and G, see Figure 1.1). In some cases, wobble pairings (G and U), or other non-canonical ones can be found. This arrangement of two nucleotides binding together is called a *base pair*. Contrary to DNA, base pairing usually occurs between bases of the same strand.

Definition 1.1.1 (Set of Base Pairs)
The **set of base pairs** *is given by* $\sum^{\boldsymbol{BP}} = \{A\text{-}U,\ U\text{-}A,\ C\text{-}G,\ G\text{-}C,\ G\text{-}U,\ U\text{-}G\}$.

1.1.1 Primary Structure

The simplest way to describe an RNA is just to specify the sequential order of its bases. As already mentioned, bases are given from 5' end to 3' end.

Definition 1.1.2 (Set of Bases)
The four element **set of bases** *is given by* $\sum^{\boldsymbol{B}} = \{A,\ C,\ G,\ U\}$.

Definition 1.1.3 (Primary Structure, Primary Sequence)
The sequence $S = S_1 ... S_n$ *of* n *bases* S_i*, where* $1 \leq i \leq n$ *and* $S_i \in \sum^B$*, is called the* **primary structure** *or the* **(primary) sequence** *of the RNA.*

Exemplarily, the primary structure of the yeast phenylalanine tRNA (tRNA$^{\text{Phe}}$) can be given by

```
5'-GCGGAUUUAGCUCAGUUGGGAGAGCGCCAGACUGAAGAUCUGGAGGUCCUGUGUUCGAUCCACAG
AAUUCGCACCA-3'
```

1.1.2 Secondary Structure

The activity of RNA is determined by its structure, the way it folds back on itself. The structure that arises after forming hydrogen bonds between bases of an RNA is called the *secondary structure* of that RNA. It is determined as the set of Watson-Crick, wobble, and other (non-canonical) base pairings that form when the RNA is folded. Here, we define a non-canonical base pair as non-Watson-Crick and non-wobble pair.

Definition 1.1.4 (Secondary Structure)
The **secondary structure** *of an RNA sequence* $S = S_1 ... S_n$ *is defined as a set* $T = \{(S_i, S_j) | 1 \leq i < j \leq n; S_i \text{ and } S_j \text{ pair}\}$ *of base pairs between nucleotides.* (S_i, S_j) *or in short* (i, j) *denotes the base pair between the nucleotides* S_i *and* S_j*. Each nucleotide can at most pair with one other nucleotide. Thus, two base pairs* (i, j) *and* (k, l) *are either identical* ($i = k$ *and* $j = l$) *or differ in both nucleotides* ($i \neq k$ *and* $j \neq l$).

As an example, the secondary structure of the yeast tRNA$^{\text{Phe}}$ is shown in Figure 1.2.

Definition 1.1.5 (pseudoknot-free)
An RNA secondary structure T *is called* **pseudoknot-free** *if for every two base pairs* (i, j) *and* (i', j') *in* T *with* $i < i'$ *holds* $i < j < i' < j'$ *or* $i < i' < j' < j$.

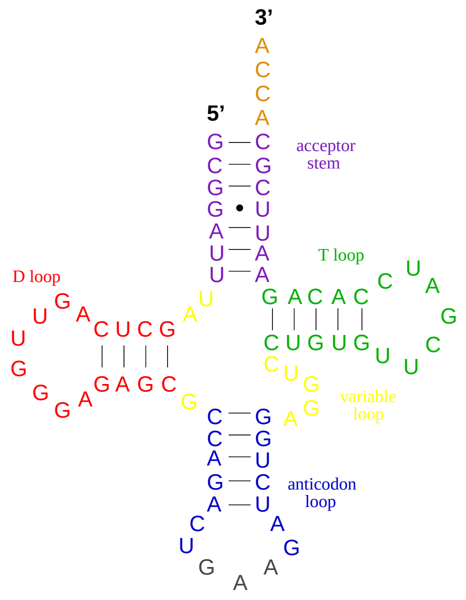


Figure 1.2: Secondary structure of yeast tRNA^{Phe} [JDR00].

Possible pseudoknots are shown in Figure 1.3. If such constructs would be taken into account, the loop decomposition (explained in the following) as well as the energy rules described in Section 1.3 break down. Thus in the following, all regarded secondary structures are pseudoknot-free. They are also called *nested structures*.

Definition 1.1.6 (Accessibility [ZMT99])
A base pair (i', j') *is called* **accessible** *from a base pair* (i, j) *if* $i < i' < j' < j$ *and if there is no other base pair* (k, l) *such that* $i < k < i' < j' < l < j$. *Likewise, a free base* i' *is called* **accessible** *from a base pair* (i, j) *if* $i < i' < j$ *and if there is no other base pair* (k, l) *such that* $i < k < i' < l < j$.

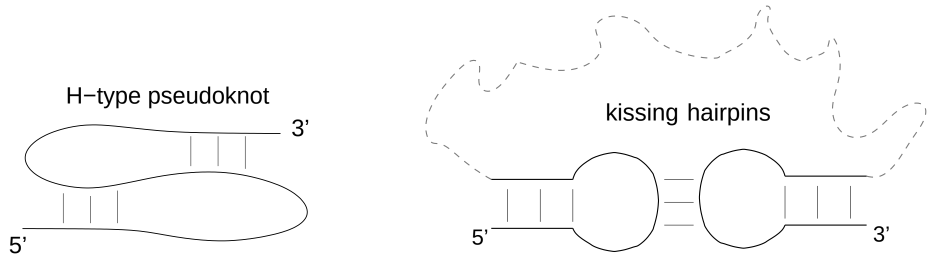


Figure 1.3: Exemplary pseudoknots.

Definition 1.1.7 (Loop, Loop Closing, Loop Inclusion [ZMT99])
The collection of bases and base pairs accessible from a base pair (i, j), but not including that base pair, is called the **loop closed by (i, j)**. *All accessible pairs and bases are also called* **included** *in the loop.*

Definition 1.1.8 (Loop Size)
The number of free bases included in a loop is also named the **size** *of the loop.*

Secondary structures can be decomposed into loops. They represent the smallest structural entities and are also called *structural elements* in the following. Depending on the number of included base pairs and free bases, we distinguish between different types of loops.

Definition 1.1.9 (Structural Elements)
The **structural elements** *a secondary structure consists of can be described as follows:*

- *A* **hairpin loop (HL)** *is a loop including only free bases and no base pair. Since sharp U-turns in an RNA secondary structure are prohibited for steric reasons, a hairpin loop must contain at least three bases.*
- *A* **stacking pair** *is a loop including only one base pair but no free bases.*
- *An* **interior loop (IL)** *includes one base pair and has a size > 0. If all free bases included in the loop are located on one side of the loop, it is called a* **bulge loop (BL)**.
- *A* **multi-branch loop (ML)** *(short: multiloop) is a loop including more than one base pair.*
- *All bases and base pairs that are not accessible from any other base pair are called* **exterior loop (EL)**.

Often, structural elements are summarized and generalized as *loops*. A visualization of the structural elements is given in Figure 1.4.

1.1.3 Tertiary Structure

For the understanding of catalytic activities of RNA, knowledge of its secondary structure alone is often incomplete. Thus, tertiary (i.e. three dimensional) atom positions and interactions have to be taken into account. As an example, the tertiary structure of the yeast tRNA^{Phe} is given in Figure 1.5. Here, the typical L-shape structure of tRNAs can be seen. The underlying secondary structure is shown in

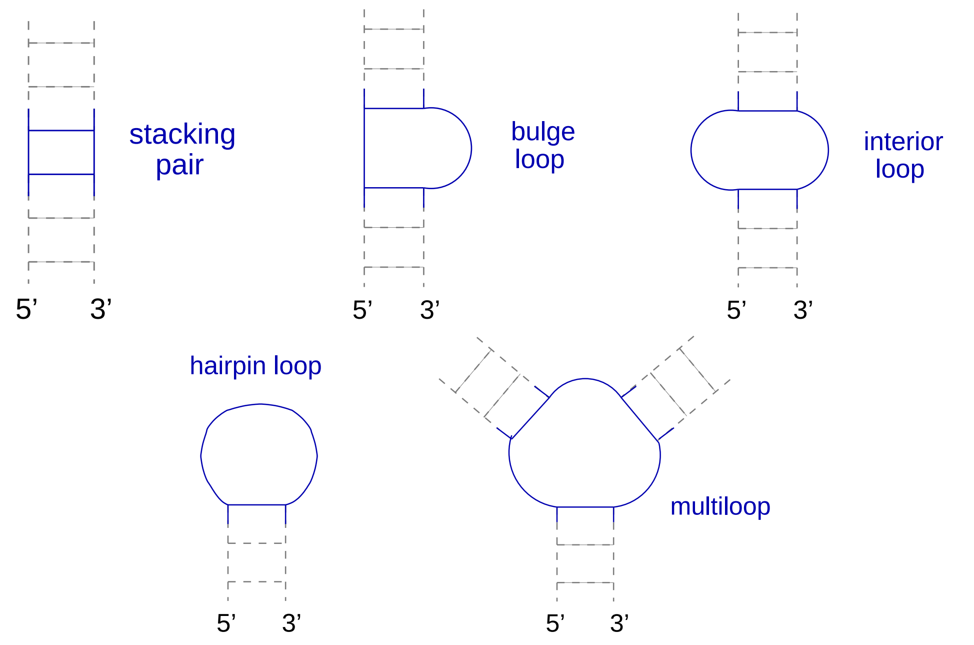


Figure 1.4: Structural elements of the RNA secondary structure.

Figure 1.2. The tertiary structure forms by additional three dimensional interactions and can be specified by its atomic coordinates. Possible tertiary interactions may include bindings where three bases are involved [LSW02].
For some RNAs, three-dimensional structures could be determined by crystallography. But RNA is hard to crystallize, which is due to their structural flexibility compared to proteins. Furthermore, secondary structure (i.e. the set of canonical and wobble base pairs) is stronger and forms faster than tertiary structure [LTM06, Woo00] and, for the most part, it determines the tertiary structure. The secondary structure can largely be identified without knowing tertiary interactions. Since prediction or experimental determination of three-dimensional RNA structures remain difficult, much work focuses on problems associated with its secondary structure. Hence in the following, we will focus on the secondary structure of RNA.

1.2 RNA Secondary Structure Representations

There are many ways to represent an RNA secondary structure, e.g. as a graph, as a string, as a rooted ordered tree, by a mountain, or a circle representation [MZS$^+$00]. Only the first two mentioned are used in this thesis.

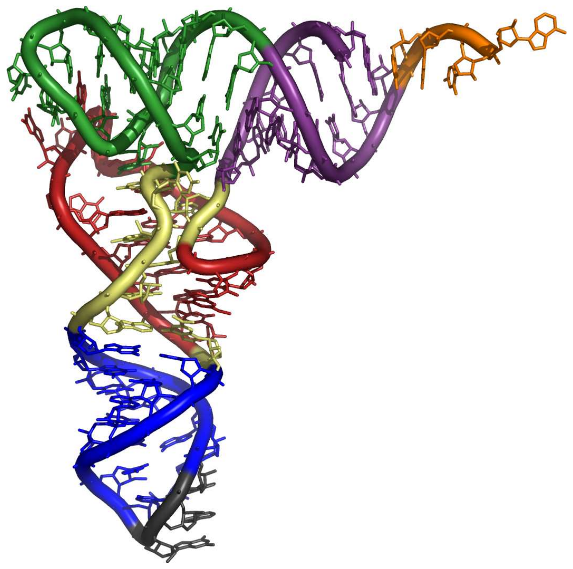


Figure 1.5: Tertiary structure of yeast tRNA$^{\text{Phe}}$. Colors are chosen according to Figure 1.2.

Definition 1.2.1 (Secondary Structure Graph [Wat78, Hof94])
A **secondary structure graph** *of an RNA sequence of length n is an undirected vertex-labeled graph with n vertices and a symmetric adjacency matrix $(A) = a_{i,j}$, where $1 \leq i, j \leq n$, fulfilling the following properties:*

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

(2) for each i, $1 \leq i \leq n$, there is at most a single $j \notin \{i-1, i+1\}$ such that $a_{i,j} = 1$ (representing base pairs).

The secondary structure graph of a pseudoknot-free RNA secondary structure also fulfills the following:

(3) if $a_{i,j} = a_{k,l} = 1$ and $i < k < j$, then $i < l < j$.

Thus, when representing the RNA secondary structure as a graph, each nucleotide is assign to a vertex. All neighbored nucleotides as well as all paired bases are connected by an edge. Figure 1.6A shows a spatial picture of the secondary structure graph, while in Figure 1.6C all vertices are arranged in one line and base pairs are represented by arcs.

The most compact representation of the secondary structure is to display it as a string of brackets and dots. Here, each base pair (i, j) is replaced by a left parenthesis '(' and a right one ')' at the ith and jth position, respectively. Unpaired bases are represented by a '.'. This representation is called *dot-bracket notation*. An example is shown in Figure 1.6B.

1.3 The Energy Model of RNA Secondary Structure

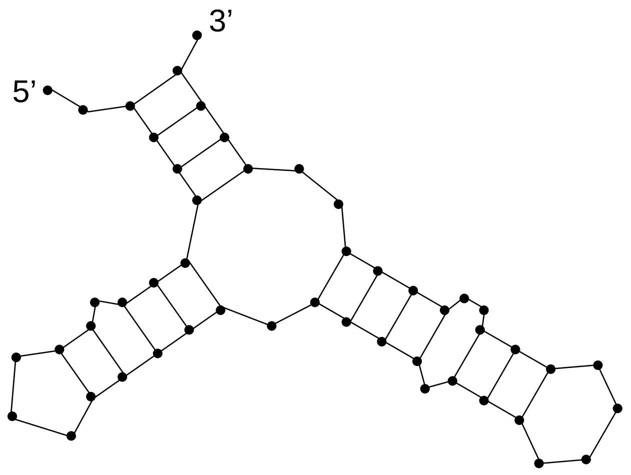
The problem of predicting the secondary structure of a given RNA sequence is called the *RNA folding problem*. Today, the most common computational approach is based on a thermodynamic model that assigns a free energy value to each secondary structure [Zuk94]. The structure with the lowest possible free energy, the *minimum free energy (mfe) structure*, is expected to be the most stable secondary structure for a given RNA sequence.
In various experimental tests, Douglas H. Turner and colleagues [FKJ+86, TSJ+87, TS88, JTZ89, MSZT99, LTM06] determined several thermodynamic parameters that can be used for computational prediction of RNA secondary structures. These are also called *nearest neighbor energy rules* since they assign free energies to loops rather than to single base pairs. Thus, a free energy is assigned to each structural element (as defined in Definition 1.1.9) independently and summed up additively to the free energy $e(T|S)$ of the complete RNA sequence S folded into the RNA secondary structure T, which can be determined by

$$e(T|S) = \sum_{loop \in T} e(loop) \tag{1.1}$$

where $e(loop)$ is the free energy of a structural element. The energy contribution of a structural element depends on its type. Generally, negative stabilizing energies are assigned to stacking base pairs and unpaired bases adjacent to a base pair, whereas destabilizing energies are associated with bulge, interior, hairpin, and multiloops [Zuk94]. The energy contributions of the different structural elements in the model according to Zuker, Mathews, and Turner [ZMT99] are explained in the following.

Stacking pair. Free energy values for all 21 possible combinations of two base pairs from $\sum^{BP}$ were determined experimentally. They do not depend on anything else. A group of two or more consecutive base pairs is called a *helix*.

A)



B)

5’ ..(((((((.((...))))).((((.(((....)))..))))..)))). 3’

C)

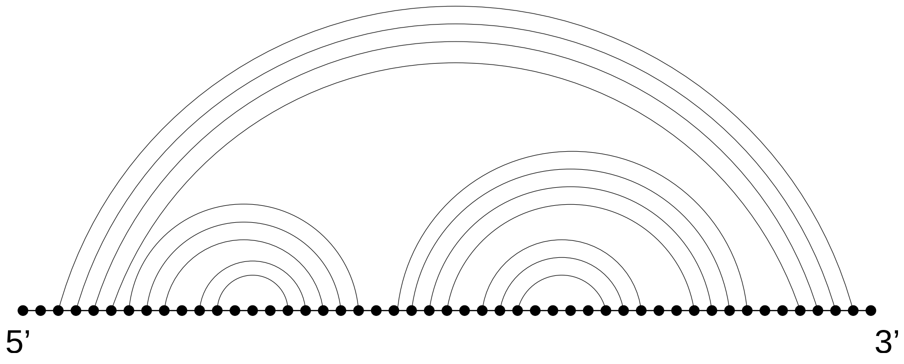


Figure 1.6: Different representations of an RNA secondary structure used in this thesis. A) shows the widely used representation as secondary structure graph. Vertex labels are left out for clarity reasons. B) represents the structure in dot-bracket notation. C) displays another version of the RNA secondary structure graph shown in (A).

The energy contributions of all kinds of **loops** (HL, BL, IL, EL, and ML) have in common that they ignore the assignments of the unbound bases included in the loops (except for unbound bases adjacent to included or closing base pairs of the loops). This is due to the current inability to quantify these effects experimentally. Thus, they are estimated by the number of free bases in the loops in order to predict the stability of any possible sequence [MSZT99].

Hairpin loop. The free energy contribution of a HL depends on three terms:

1. a size-dependent parameter,
2. for loops of a size larger than three, a *hairpin loop terminal stacking energy* is added due to the interaction between the closing pair and both adjacent unpaired bases, and
3. for specific tetraloops (HLs of size four), special *bonus energies* are added.

Bulge loop. Free energies for BLs depend basically on the loop size. Furthermore for bulge loops of size one, the stacking contribution of the closing base pair and the pair included in the loop is added.

Interior loop. Similarly to the hairpin loop, the free energy fraction of an IL depends on three items:

1. a size-dependent value,
2. *interior loop terminal stacking energies* are added for all unpaired bases adjacent to the closing pair and to the included pair, respectively, and
3. an *asymmetric loop penalty* is added for interior loops if the left loop size differs from the right loop size.

Exterior loop. The only free energy contribution of an EL arises from free bases in the loop that are adjacent to base pairs of the loop. This fraction is called *dangling base energy*. If a free base is adjacent to two base pairs, the favorable (i.e. smaller) dangling base energy is added. Free bases that are not adjacent to a base pair do not contribute to the energy.

Multiloop. Basically, the energy fraction of a multiloop is determined by a term $e_{\mathrm{ML}}(s, f)$ that depends on the number of free bases f in the multiloop as well as on the number of stems s that originate from the loop (i.e. included base pairs) plus the closing pair [ZMT99]. The following equation gives an estimation of this energy fraction, which was also made due to the current inability of quantifying these effects experimentally depending on any possible assignment of the bases.

$$e_{\mathrm{ML}}(s, f) = a + b * f + c * (s + 1) \tag{1.2}$$

where a, b, and c are constants representing an *offset*, a *free base penalty*, and a *helix penalty*, respectively. Additionally, dangling base energies have to be added similar to exterior loops.

Furthermore, we distinguish between GC and non-GC closing base pairs. A penalty (called *terminal AU penalty*) is assigned to all non-GC closing pairs at the end of a helix or a hairpin loop of size three (called *triloop*). This penalty is also added in case of included and closing non-GC pairs of bulge loops of sizes larger than one.

Widely used folding algorithms applying these energy parameters are based on dynamic programming and find a nested structure having minimum free energy within $O(n^4)$ time (given a sequence of length n) [ZS81, LZP99]. The most frequently used implementations are *mfold* [Zuk94, Zuk03] and *RNAfold* [HFS$^+$94]. They restrict the maximal size of interior loops and thus find the minimum free energy within $O(n^3)$ time.
Even if various experimental data are available for parameterization nowadays, they might still be not precise enough. Small changes in energy parameters can result in large changes in predicted foldings. Thus, the problem is 'ill-conditioned' in a mathematical sense [ZJT91]. Furthermore, it is not clear whether the real RNA molecule folds to the structure having the lowest free energy or to a different one that has slightly higher energy. Indeed, the real biological structure is often contained in the set of few suboptimal structures. This might happen if the RNA molecule traps in a local minimum of the energy landscape during the folding process. Therefore, kinetic folding algorithms try to simulate the folding process [Mar84, MDK85, FFHS00].

Additionally to the thermodynamic and the kinetic approaches, which are both energy directed, RNA secondary structure can be predicted by phylogenetic comparison. Here, a large number of sequences having similar secondary structures is needed. They must neither be to similar nor to divergent [Hof94]. The most prominent strategies are (i) the generation of a multiple sequence alignment and a subsequent determination of the consensus secondary structure inferred from the evolutionary and energetic information contained in the alignment and (ii) the simultaneous identification of the alignment and the consensus structure (called the 'Sankoff algorithm') [San85, HBS04, GG04, WRH$^+$07].

In recent years, further phylogenetic methods were developed. Stochastic context-free grammars (SCFGs) have been shown to be a possible methodology for modeling of RNA structure [KH99, KH03]. Nevertheless, the best energy-directed methods perform still better than the best SCFG-orientated approaches [DE04].

In 2006, CONTRAfold, a new secondary structure prediction tool using a flexible probabilistic model called conditional log-linear model (CLLM), was introduced. CLLMs incorporate both; the computational parameter learning of SCFGs and some complex scoring schemes used in energy-based structure prediction. Using CONTRAfold, the authors obtained the highest single sequence prediction accuracies to date [DWB06].

However, the thermodynamic approach of finding the structure having the lowest free energy is widely used. This is due to its usability without knowing any other sequences and structures. The only required prior knowledge is the set of energy parameters previously determined by Douglas H. Turner and colleagues [FKJ$^+$86, TSJ$^+$87, TS88, JTZ89, MSZT99, LTM06]. Furthermore by using the thermodynamic approach, one can calculate the partition function of a sequence and its ensemble of possible structures [McC90], which is not possible with other secondary structure prediction approaches. Additionally, the probability of each structure can be determined based on the partition function.

In the following, all introduced algorithms are based on the thermodynamic model introduced in this section. Its additive decomposition is its main advantage and the reason why we are using it. All used energy parameters are downloaded from Michael Zuker's homepage on
`http://frontend.bioinfo.rpi.edu/zukerm/cgi-bin/efiles-3.0.cgi`.

Chapter 2

RNA Design - The Inverse Folding of RNA

2.1 Biological Introduction and Importance of the Problem

Since the function of RNA molecules depends crucially on their structure, the design of RNAs having special structural properties is of interest for biologists in several fields. RNA molecules that catalyze others are called ribozymes. They are involved in the regulation of RNAs, most commonly in the cleavage of an RNA or DNA strand. Here, the group I self-splicing introns are a widely known example. The cleavage at the correct site requires a special RNA secondary structure [DS89, Cec92]. Other described ribozymes are the self-cleaving hammerhead and hairpin ribozymes and the *trans*-cleaving ribonuclease P (RNase P). The latter is involved in the processing of tRNAs, while the former two are involved in gene control [SP07]. This catalytic variety of RNA opens the question of the design of new ribozymes with possibly new catalytic functions. Speaking of RNA design, we always refer to the design of an RNA sequence that folds into a functional structure.
The design of new ribozymes may support the design of drugs and human therapeutic agents, respectively [UM95]. For example, the production of better agents for the inhibition of gene expression might be used to understand biology or for pharmaceutical applications. Several ribozymes have a catalytic core that assures their activities. Thus, RNA design may help to define the minimum structural requirements for catalysis [DS89, BJ90, Cec92].
Developing new systems with increasing functional density requires the understanding of the design of molecular structures. For the construction of nanoscale devices with novel mechanical or chemical functions, nucleic acids seem to be an excellent medium [DLWP04].

Finally, the design of RNA molecules having special functions can facilate the research on the function of natural occurring RNAs [AFH$^+$04]. It may help to understand the mechanisms of catalysis and principles of RNA folding [Cec92] and allows to predict novel sequences that are functionally equivalent but unrelated to naturally occurring RNAs [Hof94].

Based on all these findings, here, we consider the *inverse RNA folding problem*, which is the design of RNA sequences that fold into a desired structure. Apart from its application to ribozymes and riboswitches [Kni03, WNR$^+$04, Cec04] mentioned above, it can be applied to the design of non-coding RNAs, which are involved in a large variety of processes, e.g. gene regulation, chromosome replication, and RNA modification [Sto02].
Furthermore, the inverse RNA folding allows the design of cis-acting mRNA elements such as the iron responsive element (IRE) and the polyadenylation inhibition element (PIE). Both elements have a conserved secondary structure and few conserved sequence positions in loops. By providing binding sites for regulatory proteins, they determine mRNA stability and translation efficiency. The IRE is essential for the expression of proteins that are involved in the iron metabolism [HK96]. The PIE contains two binding sites for U1A proteins [VGM$^+$00]. U1A binding leads to an inhibition of the poly(A) polymerase and a reduced mRNA stability and translation efficiency due to a shortened poly(A) tail.

2.2 Existing Approaches and their Limitations

Given an RNA secondary structure, we aim at finding an RNA sequence that adopts this structure. Only compatible sequences (see Definition 2.3.2) are considered as candidates in the inverse folding procedure. Clearly, a compatible sequence can but need not have the target structure as its minimum free energy (mfe) structure. It is impossible to test each compatible sequence, whether its mfe structure is the searched one, since the number of sequences grows exponentially in the size of the structure [Hof94]. The exact complexity of the design problem is unknown, in particular it is not known whether a provable efficient (i.e. polynomial-time) algorithm exists for the inverse RNA folding problem [AFH$^+$04, AHHC07]. Thus, different heuristic local search strategies, which do not analyze the complete solution space, were used by existing programs dealing with inverse RNA folding.

Nevertheless, there are only few publications that address the inverse RNA folding problem, e.g. [Hof94, AFH$^+$04, DLWP04]. One approach is implemented in RNAinverse, which is included in the Vienna RNA Package [Hof94]. They use the strategy of adaptive walk and find local optima concerning a structure distance between the mfe structure of the designed sequence and the target structure (*mfe-mode*), or concerning the probability of folding into the target structure (*p-mode*). During the adaptive walk, RNAinverse starts with a random sequence and finds new candidate sequences by iteratively modifying single unpaired bases or base pairs in order to find a sequence that fits better their optimization criterion (structure distance or probability). As soon as they have found a better sequence, the modified position(s) are mutated and the search for better candidates continues. It stops, if a solution is found (a sequence whose mfe structure is the target one) or no modification provides a better candidate, which means, that the search is stuck in a local optimum. During the mfe-mode, the adaptive walk is done recursively since substructures contribute additively to the energy. Small substructures are optimized first and proceeded to larger ones. RNAinverse acts very well, if we consider short structures (up to 200 bases). But for larger structures, it is quite slow and maximizing the probability of folding into the target structure does not work anymore since the adaptive walk is not done recursively during p-mode. There, the optimization is done on the whole sequence and thus, too much computation time is needed.

Using RNAinverse, Schuster *et al.* [SFSH94] derived some thousand sequences for several structures. They found that their pairwise sequence distances are not distinguishable from the distances of random sequences compatible with the given structure. This finding supports the hypothesis that sequences folding into the same structure are randomly distributed in the sequences space and thus, there is no need to search large fractions of the sequence space to find a sequence folding into the given structure. They further stated that the average number of mutations, which are needed to convert a random compatible sequence into one that folds into the target structure, is much smaller than the sequence length. In contrast, Andronescu *et al.* [AFH$^+$04] as well as ourselves [BB06] found several structures that are hard to design independent of their sizes. These findings support the difficulties in finding the correct complexity of the design problem.

Dirks *et al.* [DLWP04] have also used an adaptive walk to analyze several objective functions (values to be optimized) and compared them to each other. They found out that (in case of adaptive walk) the most successful objective functions are maximizing the probability of folding into the desired structure and the newly introduced function of minimizing the average number of incorrect paired nucleotides. Unfortunately, they gave no hint concerning time needed to compute the solutions.

Furthermore, Andronescu *et al.* [AFH+04] have developed an algorithm called RNA-SSD (RNA Secondary Structure Designer). It is based on a recursive stochastic local search, which tries to minimize a structure distance of the target structure and the mfe structure of the designed sequence. RNA-SSD also uses the fold functions of the Vienna RNA Package. In a first step, RNA-SSD creates a starting sequence whose mfe structure is close to the target one. To do so, RNA-SSD uses different probabilistic models for paired and unpaired positions when assigning bases randomly to the sequence. Furthermore, their algorithm avoids complementary stretches of bases except they are desired along two sides of a stem [AFH+04]. During a second step, they use a hierarchical decomposition of the structure into smaller substructures to reduce the complexity of the problem. Then, they apply a stochastic local search (SLS) to the smallest substructures and finally combine them to larger ones. Similar to the adaptive walk, the SLS finds new candidate strands by iteratively modifying single unpaired bases or base pairs. The modified position(s) are mutated if a better sequence is found. If the candidate sequence has a worse value concerning the objective function, it is mutated with a low probability. The search is stopped, if a solution is found (a sequence whose mfe structure is the target one) or a maximum number of mutations is done. RNA-SSD is available online, but the size of the input structures is restricted to 500 there.

Using an advanced version of RNA-SSD, Aguirre-Hernández *et al.* [AHHC07] propose that the empirical time-complexity of the RNA design problem is polynomial. They estimated a median expected run-time of about $O(n^3)$ for RNA-SSD and of about $O(n^5)$ for RNAinverse, where n is the size of the structure.

Westhof *et al.* [WMJ96] proposed the construction of a combinatorial library consisting of modular units, which can be used for the creation of new RNA molecules with a given structure. This approach is based on the hierarchical organization of the folding process, i.e. secondary structure elements form first while tertiary contacts are composed afterwards. According to our experiments, constructing RNA sequences from small units is not successful in most cases (data not shown).

2.3 A New Approach: INFO-RNA

Here, we present a new algorithm for the design of RNA sequences that fold into a given structure. Since the problem of finding the secondary structure of a given RNA sequence is called the RNA folding problem, the inverted case is called the *inverse RNA folding problem*. Formally, it can be defined as follows.

Definition 2.3.1 (Inverse RNA Folding)
The **inverse folding of RNA** *is the problem of finding an RNA sequence $S = S_1...S_n$ of length n that folds into a given secondary structure T, where $S_i \in \sum^B = \{A, C, G, U\}$ for $1 \leq i \leq n$. T can be described as a set of pairs as defined in Section 1.1.2.*

Definition 2.3.2 (Compatible Sequence)
A sequence is called **compatible** *to a given structure, if it can form all required base pairs regardless of energy. The set of all compatible sequences is called $\boldsymbol{\mathcal{S}}$.*

To find the best sequence that folds into a given structure, a search space of an exponentially high number of compatible RNA sequences has to be analyzed. Therefore, it takes exponential time to find a globally optimal solution by testing all candidate sequences and thus, local search methods are widely used to address the inverse folding problem. Consequently, the resulting local optima are not guaranteed to be globally optimal but are optimal among all their *sequence neighbors*.

Definition 2.3.3 (Sequence Neighbor)
All compatible sequences that differ from a sequence S in one unbound position or in two positions, which have to pair in structure T, are called **sequence neighbors** *of sequence S (see Figure 2.1).*

We introduce a new algorithm *INFO-RNA* attending the INverse FOlding of RNA. It consists of two steps; a new design method for good initial sequences and a following improved stochastic local search that uses an effective neighbor selection method.
Except on the search strategy itself, the performance of the local search depends on the quality of the initializing sequence. Often, it is chosen at random. We found that a good choice is to use a sequence that among all sequences adopts the given structure with the lowest possible energy. We present a dynamic programming approach to solve this problem. Here, multi-branched loops are especially complicated to handle.
In the following, we introduce the new method to create an excellent initializing sequence and describe the subsequent local search strategy.

2.3.1 The Initializing Step

The initializing step of INFO-RNA uses the technique of dynamic programming. This method was successfully applied to RNA secondary structure prediction as mentioned in Section 1.3 [ZS81] and related problems. Similar to Zuker and Stiegler [ZS81], we use free energies of structural elements [stacks, bulge- (BL), interior- (IL), hairpin- (HL), multiloops (ML)] as defined in Section 1.3. They

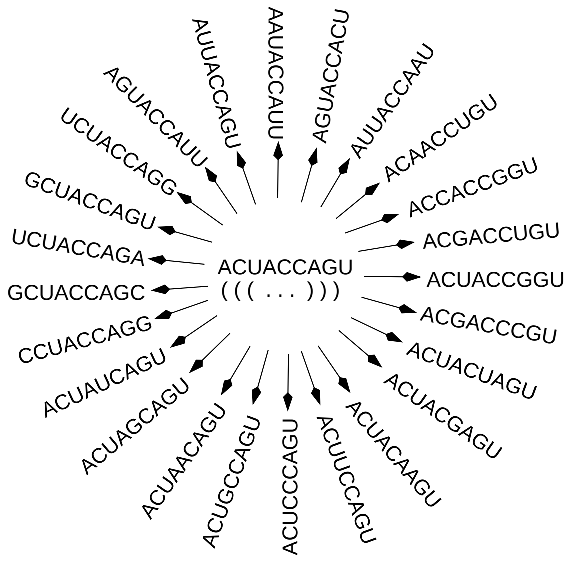


Figure 2.1: Exemplary sequence (and structure) and all its sequence neighbors that can adopt this structure. The dot-bracket notation is used for the representation of the RNA secondary structure.

depend on the size of the loop, the closing and included base pairs, as well as on the free bases inside the loops that are adjacent to the closing and included pairs. Free bases that are not adjacent to a base pair do not give any energy fraction. Since each pair belongs to two elements, neighbored elements in a structure are linked and base pairs cannot be handled independently. The free energy value of a pseudoknot-free structure is calculated by adding up all partial energies of its elements (see Equation 1.1).

Given a target structure T, we find among all sequences a sequence S that adopts T with the lowest possible energy. Formally, this means that we find a sequence S resulting from

$$\operatorname*{argmin}_{S'} e(T|S') \tag{2.1}$$

where $e(T|S')$ represents the free energy of sequence S' folded into structure T. For solving this problem, our dynamic programming algorithm needs linear time depending on the structure size. It divides the target structure into its structural elements. Before explaining the algorithm, some formal definitions concerning the structural elements and the ordering of the base pairs are needed.

Definition 2.3.4 (Substructure)
A **substructure** $\mathbf{T_{(i_1,i_2)}}$ *of structure T is defined as a structural part of T that is closed by pair (i_1, i_2) and has a connected backbone (see Figure 2.2B).*

$\mathbf{e(T_{(i_1,i_2)}|(S_{i_1}, S_{i_2}) \rightarrow (a_1, a_2))}$ *is defined to be its minimum free energy under the condition that the sequence positions (S_{i_1}, S_{i_2}) of the closing pair (i_1, i_2) of the substructure are fixed to a base pair assignment (a_1, a_2).*

In the following, we will give a formal definition of a structural element, which was already introduced in Section 1.1.2 (Definition 1.1.9) in a more descriptive and informal way.

Definition 2.3.5 (Structural Element)
$\mathbf{T}^{(\cdot,\cdot)\ldots(\cdot,\cdot)}_{\mathbf{(i_1,i_2)}}$ *indicates a structural element of structure T that is closed by the pair given in the subscript (i_1, i_2) and includes all base pairs indicated in the superscript. The structural element does not need to have a connected backbone (see Figure 2.2C).*

$\mathbf{e(T^{(j_1,j_2)\ldots(p_1,p_2)}_{(i_1,i_2)}|(S_{i_1}, S_{i_2}) \rightarrow (a_1, a_2), (S_{j_1}, S_{j_2}) \rightarrow (b_1, b_2), \ldots, (S_{p_1}, S_{p_2}) \rightarrow (r_1, r_2))}$ *is defined as the minimum free energy of the structural element that is closed by base pair (i_1, i_2) and includes pairs (j_1, j_2),...,(p_1, p_2), whose sequence positions are fixed to assignments (a_1, a_2), (b_1, b_2), ..., and (r_1, r_2), respectively.*

Note, that we will use the following conventions:

- In case of a stack, a BL, or an IL, the structural element includes exactly one base pair and is closed by exactly one pair. Thus, one pair is indicated in the subscript and in the superscript, respectively.
- In case of a ML, the structural element includes more than one base pair, whose are indicated in the superscript.
- In case of an EL, the structural element has no closing pair and thus an empty subscript.
- In case of a HL, the structural element has no included pair and thus an empty superscript. Therefore, a HL can also be interpreted as a substructure.

Furthermore, we have to define an *order* $\prec$ of the base pairs in the structure. It specifies the order in which base pairs are analyzed during the initializing step of INFO-RNA.

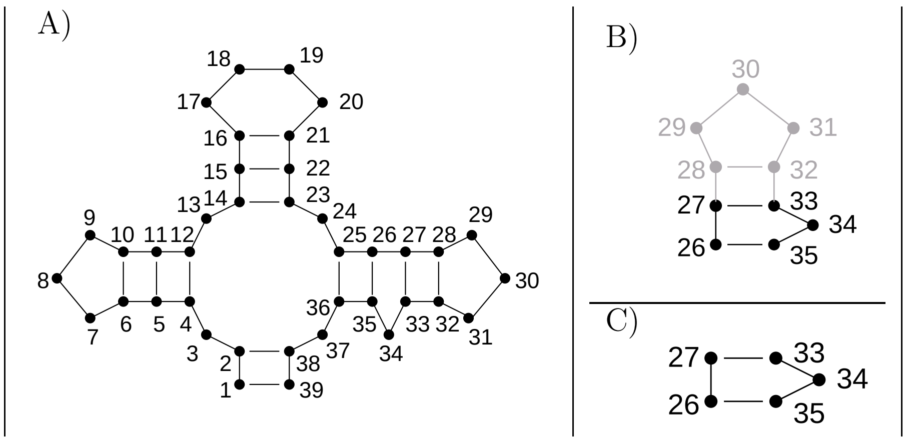


Figure 2.2: A) RNA secondary structure, B) Substructure $T_{(26,35)}$ having a connected backbone, C) Structural element $T^{(27,33)}_{(26,35)}$ without a connected backbone

Definition 2.3.6 (Base Pair Order)
All base pairs of a structure T are analyzed in a predefined **order** $\prec$, *where $(i_1, i_2) \prec (j_1, j_2)$ means that base pair (i_1, i_2) is analyzed prior to base pair (j_1, j_2). The actual order in which the base pairs are examined is defined as follows.*

$$(i_1, i_2) \prec (j_1, j_2) \quad \textit{if and only if} \quad i_1 > j_1 \tag{2.2}$$

Relating to the example of Figure 2.2A, the order of all pairs is the following:

$(28, 32) \prec (27, 33) \prec (26, 35) \prec (25, 36) \prec (16, 21) \prec (15, 22) \prec (14, 23) \prec (6, 10) \prec (5, 11) \prec (4, 12) \prec (2, 38) \prec (1, 39)$.

Notation 2.3.7 (Predecessor)
According to the definition of the order, all base pairs accessible from a fixed pair are smaller than itself and thus called its **predecessors**.

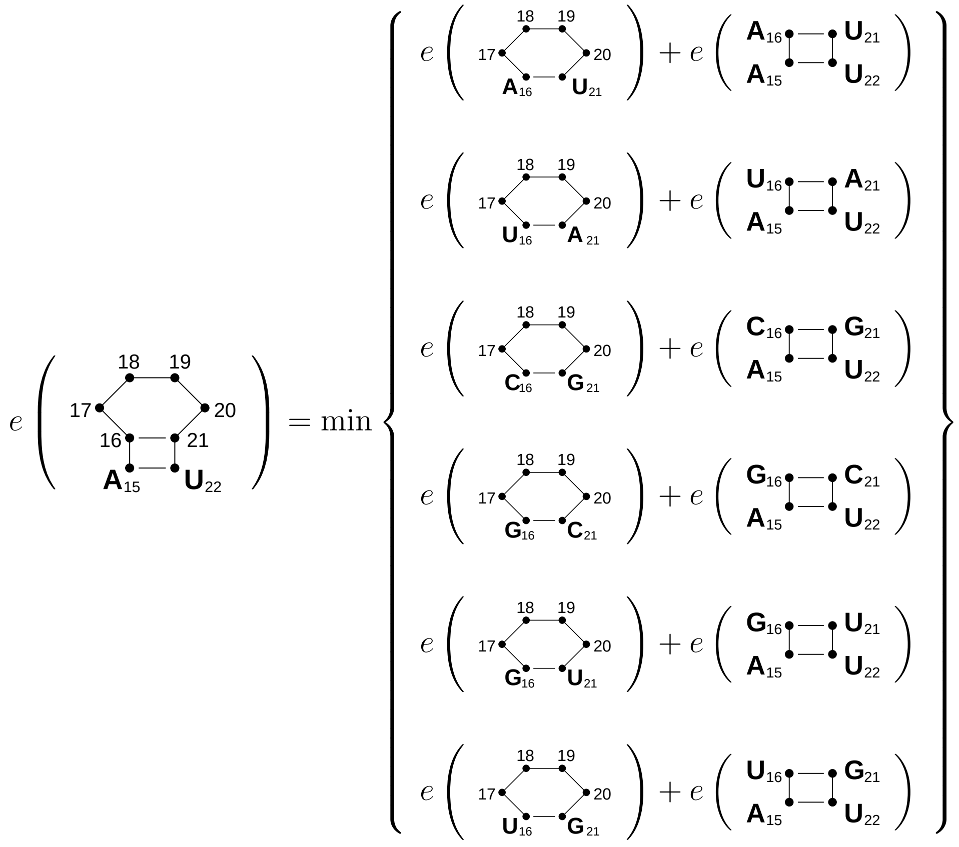
Since closing pairs of HLs have no accessible base pairs, they have no predecessor. The closing pair of a ML has as many predecessors as accessible base pairs. All other pairs have exactly one predecessor. Table 2.1 shows the predecessors of Figure 2.2A.

base pair	predecessor(s)
$(28, 32)$	none
$(27, 33)$	$(28, 32)$
$(26, 35)$	$(27, 33)$
$(25, 36)$	$(26, 35)$
$(16, 21)$	none
$(15, 22)$	$(16, 21)$
$(14, 23)$	$(15, 22)$
$(6, 10)$	none
$(5, 11)$	$(6, 10)$
$(4, 12)$	$(5, 11)$
$(2, 38)$	$(4, 12)$, $(14, 23)$, $(25, 36)$
$(1, 39)$	$(2, 38)$

Table 2.1: Base pairs and their predecessors in the structure of Figure 2.2A.

Idea. The basic idea of the initializing step of INFO-RNA is to start with small substructures and enlarge them gradually by one base pair. Thus, the algorithm starts at the closing pair of a hairpin loop, subsequently fixes it to pair assignments out of the set of valid pairs $\sum^{BP} = \{$A-U, U-A, C-G, G-C, G-U, U-G$\}$, and assigns the unbound positions of the loop such that they provide the lowest possible free energy value for this small substructure under the condition that the closing pair is fixed. This is stored for all six possible assignments of the pair.

Afterwards, the next pair to the HL-closing one is fixed. The energy can be calculated by the sum of the energy of the hairpin loop including the closing pair and the stacking energy of the current pair and the closing one of the HL. To find the best energy value, we have to minimize this sum over all possible assignments of the base pair closing the HL. This is demonstrated in Equation 2.3 exemplarily, where $e(.)$ represents the minimal free energy. The substructures refer to Figure 2.2A.


e
18 19
17 20
16 21
A15 U22
= min
e 18 19 17 20 A16 U21 + e A16 U21 A15 U22
e 18 19 17 20 U16 A21 + e U16 A21 A15 U22
e 18 19 17 20 C16 G21 + e C16 G21 A15 U22
e 18 19 17 20 G16 C21 + e G16 C21 A15 U22
e 18 19 17 20 G16 U21 + e G16 U21 A15 U22
e 18 19 17 20 U16 G21 + e U16 G21 A15 U22

(2.3)

Equation 2.4 formalizes the example of Equation 2.3.

$$
e\left(\, T_{(15,22)} \,\middle|\, (S_{15}, S_{22}) \rightarrow (A, U) \,\right) =
$$

$$
\min_{(a_1,a_2)} \left\{ \begin{array}{cc} & e\left(\, T_{(16,21)} \,\middle|\, (S_{16}, S_{21}) \rightarrow (a_1, a_2) \,\right) \\ + & \\ & e\left(T^{(16,21)}_{(15,22)} \,\middle|\, \begin{array}{l} (S_{16}, S_{21}) \rightarrow (a_1, a_2) \\ (S_{15}, S_{22}) \rightarrow (A, U) \end{array} \right) \end{array} \right\} \tag{2.4}
$$

Generally, the energy value of any structural element depends on the assignment of the base pairs and, in case of a loop, on the included unpaired bases adjacent to the pairs and on the loop size. The minimal energy of a substructure $T_{(i_1,i_2)}$

base pair	1	2	3	4	5	6
	A−U	U−A	C−G	G−C	G−U	U−G
1						
2						
3						
4						
⋮						

Figure 2.3: Dynamic programming matrix D

can be evaluated by adding the minimum energy of the one pair smaller substructure $T_{(i_1+k,i_2-l)}$ and the energy of the structural element $T^{(i_1+k,i_2-l)}_{(i_1,i_2)}$. Therefore, an already analyzed smaller substructure can be seen as black box, except for its closing pair. We calculate the lowest possible energies for substructures gradually by adding the next pair to a smaller substructure. In case of a ML, as many smaller substructures as included pairs are in the loop have to be added.

Having set the order of the pairs, a dynamic programming matrix D is filled with minimal free energies. Each row in D represents a base pair of the target structure while each column stands for a possible assignment of the base pairs (see Figure 2.3). Thus, D has as many rows as pairs are in our desired structure and six columns, which represent the assignments A-U, U-A, C-G, G-C, G-U, and U-G. The rows are sorted according to $\prec$.

In the following, pairs are no longer represented by their pairing positions, e.g. (i_1, i_2), but only by their row numbers in D. The values in the matrix $D(i, a)$ give the minimal free energy of a substructure ending at base pair i (represented by the row) that is assigned to $a \in \sum^{BP}$ (given by the column). Every substructure starts at one or more base pairs that do not have any predecessors.

Before giving a detailed description of the algorithm, we have to define some variables and notations that are used in the following equations. Now, T_i^j represents the structural element of T closed by base pair i and including base pair j, where i and j are row numbers in D. The free energy of this structural element when i and j are assigned to a and b, respectively, is given by $e\left(\, T_i^j \;\middle|\; i \to a, j \to b \,\right)$. Further definitions are shown in Table 2.2.

$p_k(i)$	k-th predecessor of pair i (sorted according to the order)
s	number of stems originating from a ML (= number of predecessors of the closing base pair of the ML)
F	number of free bases adjacent to stems in a ML
f	total number of free bases in a ML
$e_{\mathrm{ML}}(s,f)$	size-dependent energy fraction of a ML with f free bases and s stems. It equals $a + b * f + c * (s+1)$, where a, b, and c are constants (see Equation 1.2).
$e^{db}(b)$	dangling base energy of a free base assigned to b and adjacent to one or two stems in a ML or an EL
H	total number of free bases in a HL
$e_{\mathrm{HL}}(H)$	size-dependent energy fraction of a HL of size H
$e^{\mathrm{bonus}}_{a,b_1,\ldots,b_H}$	HL bonus energy depending on the assignments a and $b_1, ..., b_H$ of the closing pair and the free bases, respectively. It is lower than 0 for some special tetra HLs. Otherwise it is set to 0.
$e_{\mathrm{TM}}(a, b_1, b_H)$	terminal stacking and mismatch energy in HLs. It depends on the assignment a of the closing pair of the HL and the assignment of the directly adjacent free bases b_1 and b_H.
$e_{\mathrm{AU}}(i,a)$	terminal AU penalty. It penalizes stems, whose last pair is assigned to A and U or G and U. The same holds for base pairs that close a triloop and for base pairs included in or closing a BL of size larger than one. $e_{\mathrm{AU}}(i,a) = \begin{cases} 0.5 & \text{if } i \text{ is the last pair of a stem, the closing pair of a triloop, or included in or closing a BL of size} > 1 \text{ and } a \in \{\text{A-U,U-A,G-U,U-G}\} \\ 0 & \text{otherwise} \end{cases}$

to be continued on the next page

Table 2.2: Definition of additional terms used during the initializing step of INFO-RNA.

continuation of the previous page

L^l, L^r	left and right size of an IL or BL, respectively
$e_{\text{asym}}(i-1,i)$	asymmetric loop penalty that is added for ILs if $L^l \neq L^r$. Some exceptions for small nearly symmetric loops exist. $e_{\text{asym}}(i-1,i) = \\ = \begin{cases} 0 & \text{if } (L^l = 0) \text{ OR } (L^r = 0) \\ & \text{OR } (L^l = 1 \text{ AND } L^r = 2) \\ & \text{OR } (L^l = 2 \text{ AND } L^r = 1) \\ \min\left\{ \begin{array}{l} 3.0, \\ 0.5 * \lvert L^l - L^r \rvert \end{array} \right\} & \text{otherwise} \end{cases}$

Table 2.2: Definition of additional terms used during the initializing step of INFO-RNA.

During our dynamic programming approach, the fields in the matrix are filled row by row. Recall that the rows of D are sorted according to $\prec$. Thus, each pair i has as many predecessors as the structural element closed by i has included pairs. To compute the entries of D, we distinguish between (A) base pairs having no predecessor (closing base pairs of HLs), (B) base pairs with exactly one predecessor (closing pairs of BLs, ILs, or stacks), and (C) base pairs with more than one predecessor (closing base pairs of MLs).

(A) If base pair i has no predecessor, i.e. it is a closing pair of a hairpin loop,

$$\forall\, a \in \textstyle\sum^{BP} :$$

$$D(i,a) = e_{\text{HL}}(H) + \min_{b_1,\ldots,b_H \in \sum^B} \left\{ e^{\text{bonus}}_{a,b_1,\ldots,b_H} + \left\{ \begin{array}{ll} e_{\text{AU}}(i,a) & ,H = 3 \\ e_{\text{TM}}(a,b_1,b_H) & ,H > 3 \end{array} \right\} \right\}$$

where the minimum is taken over all possible assignments of all free bases $b_1,\ldots,b_H$ in the HL.

(B) If base pair i has exactly one predecessor, i.e. it is a closing pair of a bulge loop, of an interior loop, or of a stack,

$$\forall\, a \in \sum^{BP}:$$

$$D(i,a) = e_{\mathrm{AU}}(i,a) + e_{\mathrm{asym}}(i-1,i) +$$

$$+ \min_{b\in\sum^{BP}} \left\{ D(i-1,b) + \min_{\substack{\text{assignment of free}\\ \text{bases in } T_i^{i-1} \text{that are}\\ \text{adjacent to } i-1 \text{ or } i}} e\left(T_i^{i-1} \,\middle|\, \begin{array}{l} i \to a \\ i-1 \to b \end{array} \right) \right\}$$

where $e\left(T_i^{i-1} \,\middle|\, \begin{array}{l} i \to a \\ i-1 \to b \end{array} \right)$ gives the free energy of the structural element between pairs $i-1$ and i assigned to a and b. This energy value depends on a and b as well as on the assignment of the free bases directly adjacent to $i-1$ and i. Thus, two dependencies can be seen here: the dependency of the base pairs to each other and the dependency to the adjacent free bases.

(C) If base pair i has more than one predecessor, i.e. it is a closing pair of a multiloop,

$$\forall\, a \in \sum^{BP}:$$

$$D(i,a) = e_{\mathrm{ML}}(s,f) + e_{\mathrm{AU}}(i,a) + \qquad (2.5)$$

$$+ \min_{\substack{a_1,\dots,a_s \,\in\, \sum^{BP} \\ b_1,\dots,b_F \,\in\, \sum^{B}}} \left\{ \sum_{k=1}^{s} D(p_k(i), a_k) + \sum_{j=1}^{F} e^{db}(b_j) \right\}$$

where the minimum is taken over all possible assignments of all predecessor base pairs $a_1, ..., a_s$ and of all free bases $b_1, ..., b_F$ adjacent to them.
At first glance, this evaluation can be exponential in the number of stems originating from the ML and the number of adjacent free bases since the energy fraction of a free base adjacent to two stems depends on the assignments of both. However in nature, MLs have only a low number of stems. Thus, even the naive solution is usable in pratice.

To reduce this complexity to linear time for all MLs, we introduce a further dynamic programming matrix M resized and recalculated for each ML. It calculates the best free energy of the substructure closed by the closing pair of the ML dynamically. The evaluation of the ML starts with the first included pair according to the order of the pairs. Here, included base pairs are renumbered starting with 1 being the first included pair. Furthermore, the order is defined as given in Equation 2.2, but the definition of the predecessors is renewed.

Definition 2.3.8 (Predecessor in a ML)
Now, pair i included in a ML is **predecessor** *to pair j of the ML iff $i \prec j$ and if there is no other pair k in the ML such that $i \prec k \prec j$. The closing pair of the ML is a predecessor of the first included base pair, while the last included base pair is a predecessor of the closing pair.*

Matrix M is arranged analogously to matrix D. It has a row for each included base pair of the ML but not for the closing pair. Thus, M has as many rows as there are included pairs in the multiloop. Each column of M represents a possible assignment of the pairs, i.e. M has six columns incorporating A-U, U-A, C-G, G-C, G-U, and U-G. Hence, $M(j, a)$ gives the minimum free energy fraction of the prefix of the ML (and its originating stems) that starts at the first included base pair and ends with the free base adjacent downstream to the j-th included pair, which is assigned to $a \in \sum^{BP}$. Exemplarily, $M(2, a)$ of the ML in Figure 2.2 gives the minimum free energy fraction of the ML prefix including bases 37, 36, 25, and 24, when base pair $(14, 23)$, which is the second included pair in the ML, is assigned to a.

M has to be recalculated for each possible assignment of the closing pair since this base pair is fixed and a predecessor for the first included base pair. In each step, the best energy of the part of the ML is evaluated that includes the current base pair j and all base pairs h with $h \prec j$. To this end, all assignments of the previous base pair as well as of the stem-adjacent free base(s) between the current and the previous base pair have to be taken into account. This has to be done, since, firstly, only free bases adjacent to a base pair give an energy fraction and secondly, the energy fraction of a free base depends on the assignment of all adjacent base pairs.

Thus, we have to differentiate between the three cases of (I) no, (II) one, and (III) more than one free bases between base pairs in a ML. They are illustrated in Figure 2.4. The respective recursions are given in Equations 2.6, 2.8, and 2.10. There, j represents a base pair included in the ML and its associated assignment is denoted as a.

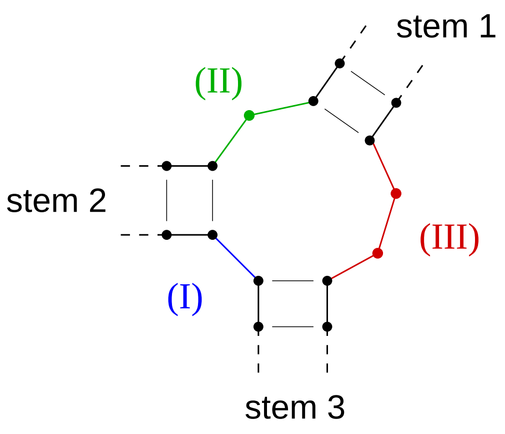


Figure 2.4: The three different cases of numbers of free bases between two consecutive base pairs in a ML. (I) corresponds to the first case, where two base pairs are neighbored to each other and thus, no free base is between them (given in blue). (II) contains the most difficult case, where a free base in a ML is adjacent to two base pairs (given in green). In case (III), more than one free bases are between two consecutive base pairs and thus, these free bases are only adjacent to one base pair each (given in red).

Notation 2.3.9 (Dangling Base Energy)
The energy fraction of a free base k in a ML or an EL that is

- *assigned to b,*
- *adjacent to base pair j assigned to a, and*
- *located upstream (up) or downstream (down) of the adjacent base in the adjacent pair*

is given by the dangling base energy $\boldsymbol{e^{db}_{up}(b,j,a)}$ and $\boldsymbol{e^{db}_{down}(b,j,a)}$, respectively.

(I) *If there is no free base between the current base pair j and its predecessor i_p*, values in M are set according to the following equations.

Basic recursive equation of case (I):

$$M(j,a) = \min_{a_p \in \sum^{BP}} \{\ M(i_p, a_p) + D(i_p^D, a_p)\ \} \tag{2.6}$$

where a_p denotes the assignment of base pair i_p. While i_p indicates the row number in matrix M, i_p^D represents its respective row number in matrix D.

Initialization of case (I):

$$M(1,a) = 0$$

Final ML recursive equation of case (I):

$$D(i,a) = e_{\mathrm{ML}}(s,f) + e_{\mathrm{AU}}(i,a) + \min_{a_p \in \sum^{BP}} \{\ M(i_p, a_p) + D(i_p^D, a_p)\ \} \tag{2.7}$$

where i_p indicates the last included base pair of the ML, which is assigned to a_p. While i_p indicates the row number in matrix M, i_p^D represents its respective row number in matrix D. Here, i represents the closing pair of the ML according to the base pair numbering of the whole structure.

In case of no free base between the last included base pair and the closing pair of the ML, Equation 2.7 replaces Equation 2.5, which is the recursive equation for finding the entry of the closing pair of the ML in matrix D.

(II) ***If there is only one free base between the current base pair j in the ML and its predecessor i_p*** *in the ML*, values in M can be evaluated using the following equations.

Basic recursive equation of case (II):

$$M(j,a) = \min_{\substack{a_p \in \sum^{BP} \\ b \in \sum^{B}}} \left\{ \begin{array}{c} \min\{e^{db}_{down}(b,j,a), e^{db}_{up}(b,i_p,a_p)\} \\ +M(i_p,a_p) + D(i_p^D, a_p) \end{array} \right\} \tag{2.8}$$

where a_p denotes the assignment of base pair i_p. While i_p indicates the row number in matrix M, i_p^D represents its respective row number in matrix D. b represents the assignment of the free base between i_p and i.

Initialization of case (II):

$$M(1,a) = \min_{b \in \sum^B} \left\{ \min\{e^{db}_{down}(b,1,a), e^{db}_{up}(b,i_c,a_c)\} \right\}$$

where i_c is the closing pair of the ML, which is assigned to a_c. Again, b represents the assignment of the free base between i_c and the first included pair in the ML.

Final ML recursive equation of case (II):

$$\begin{aligned} D(i,a) = {} & e_{\text{ML}}(s,f) + e_{\text{AU}}(i,a) + \\ & + \min_{\substack{a_p \in \sum^{BP} \\ b \in \sum^B}} \left\{ \begin{array}{c} \min\{e^{db}_{down}(b,i,a), e^{db}_{up}(b,i_p,a_p)\} \\ +M(i_p,a_p) + D(i_p^D,a_p) \end{array} \right\} \end{aligned} \tag{2.9}$$

where i_p indicates the last included base pair of the ML, which is assigned to a_p. i_p indicates the row number in M, i_p^D gives its respective row number in matrix D. Again, i represents the closing pair of the ML according to the base pair numbering of the whole structure. Once again, b represents the assignment of the free base between the closing pair i and the last included pair i_p of the ML.

In case of a single free base between the last included base pair and the closing pair of the ML, Equation 2.9 replaces Equation 2.5, which is the recursive equation for finding the entry of the closing pair of the ML in matrix D.

(III) *If there are more than one free bases between the current base pair j and its predecessor i_p*, values in M are set according to the following equations.

Basic recursive equation of case (III):

$$\begin{aligned} M(j,a) = {} & \min_{\substack{a_p \in \sum^{BP} \\ b_p \in \sum^B}} \left\{ e^{db}_{up}(b_p,i_p,a_p) + M(i_p,a_p) + D(i_p^D,a_p) \right\} \\ & + \min_{b_j \in \sum^B} e^{db}_{down}(b_j,j,a) \end{aligned} \tag{2.10}$$

where a_p denotes the assignment of base pair i_p. i_p indicates the row number in M, i_p^D gives its respective row number in D. b_p and b_j represent the assignments of the free bases adjacent to i_p and to j, respectively.

Initialization of case (III):

$$M(1,a) = \min_{b_c \in \sum^B} e^{db}_{up}(b_c, i_c, a_c) + \min_{b_1 \in \sum^B} e^{db}_{down}(b_1, 1, a)$$

where i_c is the closing pair of the ML, which is assigned to a_c. b_c and b_1 represent the assignments of the free bases adjacent to the closing pair and the first included pair of the ML, respectively.

Final ML recursive equation of case (III):

$$\begin{aligned} D(i,a) = {} & e_{\mathrm{ML}}(s,f) + e_{\mathrm{AU}}(i,a) + \min_{b_i \in \sum^B} e^{db}_{down}(b_i, i, a) + \\ & + \min_{\substack{a_p \in \sum^{BP} \\ b_p \in \sum^B}} \left\{ \; e^{db}_{up}(b_p, i_p, a_p) + M(i_p, a_p) + D(i^D_p, a_p) \; \right\} \end{aligned} \tag{2.11}$$

where i_p indicates the last included base pair of the ML, which is assigned to a_p. Again, i_p indicates the row number in M, i^D_p gives its respective row number in D. Here, i represents the closing pair of the ML according to the base pair numbering of the whole structure. b_i and b_p represent the assignment of the free bases adjacent to the closing pair i and the last included pair i_p of the ML, respectively.

In case of more than one free bases between the last included base pair and the closing pair of the ML, Equation 2.11 replaces Equation 2.5, which is the recursive equation for finding the entry of the closing pair of the ML in matrix D.

After these steps, we append an additional row to matrix D. It takes into account the dangling base energies of the free bases adjacent to the base pair(s) of the structure included in the EL. Thus, the final free energy of the structure is stored in this additional row.

Having filled the complete matrix D, we finally aim at finding the sequence that adopts the given structure with the lowest possible energy. To this end, we choose the smallest energy value of the last row of D. It gives the minimal free energy a sequence can have, when folding into the target structure. To find the sequence that provides this energy, we trace back the matrix D along the path of the best predecessor assignments. For this reason, we store traceback pointers during the computation of D. Finally, all free bases that are not directly adjacent to a base pair and thus do not give any energy value are chosen arbitrarily.

Using this dynamic programming algorithm, we obtain a sequence that among all sequences adopts the target structure with the lowest possible energy. There is no other sequence that has lower energy when folding into this structure. Nevertheless, the sequence is not guaranteed to fold into it since this sequence can have even less energy when folding into another structure. Therefore, the resulting sequence is processed further in a second step presented in the next section.

Complexity. During the initializing step of INFO-RNA, we fill matrices D and M and generate the traceback. D has six entries per row and at most $\frac{n}{2}$ many rows, since at most $\frac{n}{2}$ base pairs are possible. (To be precise, at most $\frac{n-3}{2}$ base pairs are possible since a HL has to include at least three free bases.) Thus, D consists of at most $3n$ values. Hence, we have to check what time is needed to compute one entry. For that purpose, we differentiate between the three types of entries in D depending on the corresponding base pair (types A, B, or C).

For entries corresponding to a closing base pair of a HL (type A), the assignment of all free bases in the loop is only important in case of a tetra-loop. For larger HLs, only the assignment of the free bases adjacent to the closing base pair are taken into account. Therefore, the calculation of a type A entry needs at most 4^4 steps.

During the calculation of values corresponding to pairs having exactly one predecessor (type B), the minima are taken over all assignments of the predecessor and all assignments of the included free bases that are adjacent to the base pairs. Thus, at most $6 * 4^4$ steps are needed.

By using the additional dynamic programming matrix M for closing pairs of MLs (type C), all energy values for the closing base pairs of all MLs can be analyzed in at most linear time overall. This is due to the fact that each entry in M can be evaluated in constant time. More precisely, entries of type *(I)* need at most 6 steps since they are minimized over all assignments of their predecessors. Entries of type *(II)* can be calculated in at most $6 * 4$ steps since all assignments of the predecessor base pair and the free base between the current base pair and its predecessor have to be taken into account. For type *(III)* entries, at most $6 * 4 + 4$ steps are needed. This is due to the fact that all possible assignments of the free base adjacent to the current base pair have to be considered additionally to all possible assignments of the predecessor base pair and its adjacent free base. Each matrix M has to be recalculated for every possible assignment of the closing base pair of the corresponding ML, i.e. each matrix M is calculated 6 times. Even if each ML has its own matrix M, the total number of rows in all matrices M is lower than $\frac{n}{2}$ since each base pair can only be included in a single ML.

Thus in total, all matrices M can be calculated in at most linear time, i.e. the evaluation of all type C entries in D needs in total $O(n)$ time. Furthermore, each entry in matrix D of type A or B can be calculated in constant time and hence, evaluating all of them needs $O(n)$ time, too. Consequently, the whole matrix D can be evaluated in linear time as well as the generation of the traceback.

2.3.2 The Local Search Step

After generating the initial sequence as described in the previous section, INFO-RNA found a sequence S that has the lowest free energy a sequence can have when folding into the desired structure T. However, S is not guaranteed to fold into T because it may have even less free energy when folding into another structure. Therefore, we do a subsequent local search on the sequence. The sequence is mutated iteratively in search of local optimal sequences, i.e. sequences that provide a better value than their neighbored sequences concerning a selected objective or cost function, respectively. Two *objective functions* applied in INFO-RNA are described in the following. Both of them are used in the Vienna RNA Package [Hof94] as well.

Minimum free energy structure distance (mfe-mode). When using a structure distance as objective function, the cost function to be minimized is the structure distance $d(T, T_S)$ between the desired structure T and the minimum free energy structure T_S of the designed sequence S. One of the simplest structure distance measures is the *base pair distance*, which is the minimum number of base pairs that have to be opened or closed in order to convert one structure into the other [Hof94]. A lot of other measures are conceivable (e.g. the Hamming distance, the tree edit distance, the string edit distance, etc.) but as the base pair distance is best suited for our intention (the comparison of different structures on the same sequence) [HFS$^+$94], it is used in INFO-RNA. As implemented in the Vienna RNA Package and described in [Hof94], when using the structure distance as objective function, we also use an recursive approach. The optimization starts on small substructures of T instead of working directly on the whole structure. These substructures proceed successively to larger ones. This is done since running the optimization directly on the full length sequence would take too much computation time. The idea is that a substructure, which is optimal for a subsequence, will appear in the full sequence with enhanced probability even if this is not assured.

Probability (p-mode). Using the criteria of probability, we try to maximize the probability $P(T|S)$ of sequence S folding into the desired structure T. Again, let $e(T'|S)$ be the energy of S folding into structure T'. Then the *partition function* $Q(S)$ of the structure ensemble of S is calculated as sum over the set $\Omega(S)$ of all possible secondary structures T' of S [McC90]:

$$Q(S) = \sum_{T' \in \Omega(S)} exp(-\frac{e(T'|S)}{RT_B})$$

where T_B is the temperature and R the gas constant given in kcal/mol. Accordingly, the effective free energy $G(S)$ of the Boltzmann ensemble of S is given as

$$G(S) = -RT_B \ln Q(S).$$

Thus, we can compute the probability of S folding into T at thermodynamic equilibrium by

$$P(T|S) = \frac{1}{Q(S)} \exp(-\frac{e(T|S)}{RT_B}) = \frac{\exp(-\frac{e(T|S)}{RT_B})}{\exp(-\frac{G(S)}{RT_B})} = \exp(-\frac{(e(T|S) - G(S))}{RT_B}).$$

Therefore, the maximization of this probability can be done by minimizing the cost function $c(S) = E(T|S) - G(S)$ [Hof94]. In combination with the probability criterion, all optimization steps are acting on the full length sequence. Thus, it is not recommended to apply this approach to structures larger than 200.

Both optimization criteria (mfe-mode and p-mode) can be used in INFO-RNA. Furthermore, they can be applied one after another. In [DLWP04], Dirks *et al.* introduced a further optimization criterion. They minimized the *average number of incorrectly paired nucleotides* $w(S)$, which is described in Section 3.7.1 more detailed. For INFO-RNA, it was not tested successfully and, thus, not further discussed here.

Stochastic local search. Apart from the objective function, the success of the local search depends on the search strategy itself. In INFO-RNA, the local search strategy is a *stochastic local search* (SLS). Today, stochastic local search algorithms belong to the standard methods for solving hard computational problems [Hoo98]. The SLS moves to the first found neighbor of the current sequence that has a better value according to the chosen objective function. To prevent the search from getting stuck in local optima (i.e. sequences that are better than all their neighbors but not necessarily the globally best solution), the SLS is allowed to move to worse neighbors with a fixed probability p_w. Thus, a tested neighbor is retained if its value concerning the objective function is better than the value of the current sequence. Otherwise, it is kept with probability p_w. The search terminates after a fixed number of steps. We set this number to ten times the length of the structure. As moves to worse neighbors are allowed, the last sequence is not necessarily the best one found. Thus, the best found sequence is stored during the complete search and finally given.

As in RNAinverse, we increase the efficiency of the search approach by not taking all sequence neighbors as candidates for mutation. Only sequence neighbors that differ from the current sequence in positions that (a) do not pair correctly and (b) positions adjacent to those are *candidates for mutation* since they give the best chances of improvement. This is due to the fact that only positions adjacent to a pair give a sequence dependent contribution to the free energy of the structure [Hof94].

In RNAinverse, the classification of the positions is done in the same way. There, not correctly paired positions are tested first and positions adjacent to them afterwards. Within the two groups (a) and (b), candidates are analyzed in an arbitrary order (*NA-mode*). In contrast, in INFO-RNA the order of testing the neighbors can alternatively be chosen depending on energy with a look-ahead of one selection step (*NE-mode*) or arbitrarily (*NA-mode*) as in RNAinverse. In the following, we introduce the new idea of the NE-mode.

NE-mode. To use a look-ahead of one selection step, the energy of each candidate sequence folded into the target structure T is calculated. Afterwards, the resulting energy difference to the current sequence folded into T is evaluated. Let $e(T|S)$ be the energy of sequence S folded into the wanted structure T. Let S' be a neighbor of S that is compatible to T and that is a candidate for mutation. Let $e(T|S')$ be the energy of S' when folding into T. Then, the energy difference is given by $e(T|S) - e(T|S')$. The higher the difference is, the earlier the neighbor is examined according to the actual optimization criterion. This pre-selection step can be done easily, since all structural elements contribute additively to the energy of the whole structure and thus, only structural elements that are closed by the mutated pair or include a mutated pair or free base have to be re-evaluated.

To evaluate the folding energy, we use functions from the Vienna RNA Package, since it includes the most efficient publically available version of Zuker's algorithm of which we are aware.

Symbol		Set of Nucleotides	Origin of designation
A	=	{A}	Adenine
C	=	{C}	Cytosine
G	=	{G}	Guanine
U	=	{U}	Uracil
R	=	{A, G}	puRine
Y	=	{C, U}	pYrimidine
M	=	{A, C}	aMino
K	=	{G, U}	Ketone
W	=	{A, U}	Weak interaction (2 H bonds)
S	=	{C, G}	Strong interaction (3 H bonds)
H	=	{A, C, U}	not-G, H follows G in the alphabet
B	=	{C, G, U}	not-A, B follows A
V	=	{A, C, G}	not-U, V follows U
D	=	{A, G, U}	not-C, D follows C
N	=	{A, C, G, U}	aNy

Table 2.3: IUPAC symbols for nucleic acids [CB85].

2.4 The Incorporation of Sequence Constraints

Up to now, we concentrated on the constraint-free inverse RNA folding problem. We searched for any RNA sequence that folds into a given structure. Now, we consider the *inverse RNA folding problem satisfying sequence constraints*, which is the design of RNA sequences that fold into a desired structure and fulfill some given constraints on the primary sequence. Such constraints are important, for example when designing ribozymes or tRNAs since certain base positions must be fixed in order to enable interactions with other molecules.

Definition 2.4.1 (Set of IUPAC Symbols)
The **set of IUPAC symbols** *is given by* $\sum^{\boldsymbol{IUPAC}} = \{A, C, G, U, R, Y, M, K, W, S, H, B, V, D, N\}$.

Definition 2.4.2 (Inv. RNA Folding Incorp. Sequence Constraints)
The **inverse RNA folding incorporating sequence constraints** *is the problem of finding an RNA sequence $S = S_1 ... S_n$ of length n that folds into a given secondary structure T and fulfills given constraints $C = C_1 ... C_n$ on the primary sequence, where $C_i \in \sum^{IUPAC}$ for $1 \leq i \leq n$, which means $S_i \in C_i$ for $1 \leq i \leq n$.*

Constraints on the primary sequence of RNA are given using IUPAC symbols. Generally, constraints can restrict certain positions to fixed nucleotides or to a fixed set of nucleotides, e.g. $C_i = R$ and $S_i \in C_i$ means $S_i \in \{A, G\}$.

All constraints have to be fulfilled during both steps of the algorithm. To this end, only entries of matrix D corresponding to allowed assignments of the base pairs are evaluated during the initialization. All other entries are set to $+\infty$. After finishing the initializing step, we get a sequence that adopts the target structure with the lowest free energy that is possible if the constraints are fulfilled. During the subsequent local search step, only mutations that coincide with the constraints are valid.
If the constraints on the sequence are not strictly fixed, the user can specify some positions where violations of the constraints are allowed. Furthermore, the user can restrict the maximal number of constraints $V^{\max}$ that may be violated in the final sequence. This might be useful if one allows violations of two different constraints but wants at most one violation in the finally designed RNA sequence. Finally, the best found RNA sequence satisfying the sequence constraints with a most $V^{\max}$ is given. This advanced approach is applicable to the design of RNA elements that include conserved nucleotides, which are essential for binding of proteins.

2.5 Results

First, we evaluated the performance of INFO-RNA without sequence constraints and compared it to two other approaches dealing with inverse RNA folding: RNAinverse from the Vienna RNA Package [HFS+94] and RNA-SSD [AFH+04]. For that purpose, we chose several artificially generated RNA structures as well as some test sets containing real biological data. To make our results comparable, we chose some biological test sets similar to that of Andronescu *et al.* [AFH+04]. Since the online version of RNA-SSD[1] can only deal with structures up to a length of 500, we used an executable of RNA-SSD, kindly provided by the authors, and repeated the tests they had done. This is due to two additional reasons. First, RNA-SSD was improved since its publication and second, we used faster PCs.

[1]http://www.rnasoft.ca

When using RNAinverse, maximizing the probability of folding into the target structure (p-mode) often gives better results than minimizing the structure distance between the mfe structure of the designed sequence and the target structure (mfe-mode). Unfortunately, the former works only for short structures up to a size of approximately 200 since it operates on the whole structure. Thus, we always chose the mode of RNAinverse that gives a result at all and if both modes obtained a solution, the better one was chosen. We call a run *successful*, if the mfe structure of the final sequence is the target structure. Otherwise, it is denoted as *unsuccessful*.

All computations were done on PCs with 3 GHz Intel Pentium 4 processors and 2 GB RAM. Since all tested algorithms are non-deterministic, we performed multiple runs on each problem instance. Detailed descriptions of the tests we did are given in the respective sessions. In the following, all given *runtimes* E_T denote expected times required for finding a solution. We decided to use expected values instead of average values because we are interested in the expected time until a solution is found. E_T is estimated by

$$E_T = A_S + N_U * A_U,$$

where A_S and A_U denote the average time needed for a successful and an unsuccessful run, respectively. N_U indicates the expected number of unsuccessful runs prior to the first successful run. It can be determined by

$$\begin{aligned} N_U &= (1-f_s) + (1-f_s)^2 + (1-f_s)^3 + ... \\ &= \sum_{i=1}^{\infty} (1-f_s)^i \\ &= \tfrac{1-f_s}{1-(1-f_s)} \\ &= \tfrac{1-f_s}{f_s} \\ &= \tfrac{1}{f_s} - 1 \end{aligned}$$

where f_s is the fraction of successful runs. Thus, the expected time for finding a solution results from the sum of the average time A_S needed for a successful run and the expected number of unsuccessful runs until a successful run is performed multiplied by the average time needed for an unsuccessful run A_U. Runtimes are given in CPU seconds and calculated as in Andronescu *et al.* by

$$E_T = A_S + (\frac{1}{f_s} - 1)A_U.$$

A problem arises if f_S is 0. Then the expected time E_T is set to infinity, which means that a solution will never be found. In the following tables, we indicate these cases by a dash $-$. During our tests, INFO-RNA will be used in mfe-mode in combination with the NE-mode, if nothing further is mentioned. Furthermore, we set the probability of keeping a worse neighbor p_w to 0.1 since this value has turned out to be the best value during some earlier tuning experiments.

2.5.1 Artificial Test Sets

Our first three test sets consist of artifically generated structures (see Table 2.4A). For that purpose, we generated RNA structures with user-given structural features, e.g. the overall size of the structure, loop sizes, and the length of the stems. For all sizes, minimal and maximal values are fixed. A structure generator chooses values among valid sizes as well as structural elements at random. The values we used are summarized in Table 2.4B.

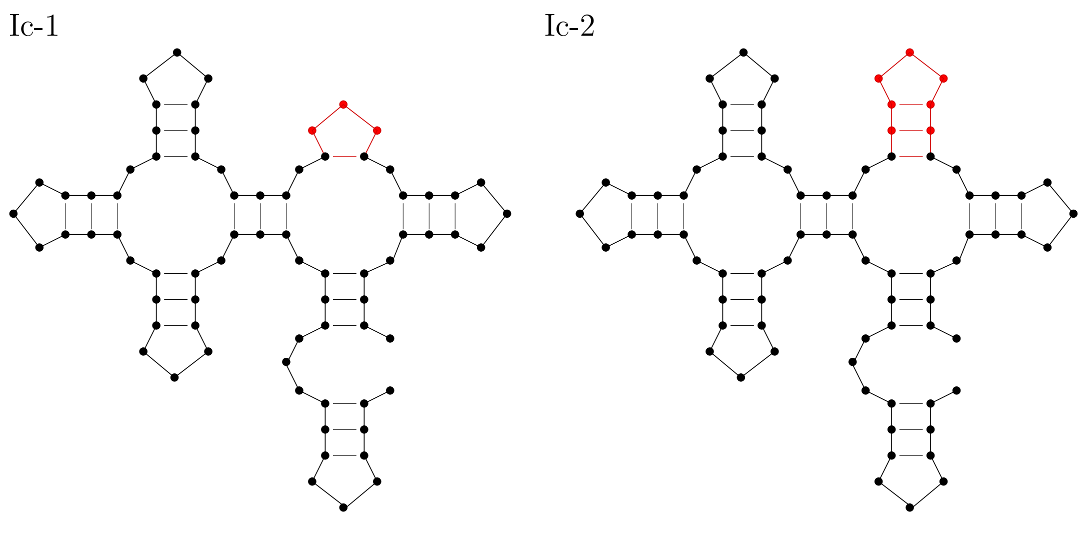


Figure 2.5: Special ML structures of Ic differing in the part given in red.

A)	Set name	#Instances	Size
	Ia	300	30 – 200
	Ib	300	300 – 700
	Ic	2	76 – 80

B)		
	stem length	3 – 10
	hairpin loop size	3 – 8
	bulge loop size	1 – 6
	internal loop size (per side)	1 – 6
	stems per multiloop (except the closing one)	2 – 5
	free bases between stems in a multiloop	0 – 6
	stems per exterior loop	1 – 2
	free bases between stems in an exterior loop	0 – 3

Table 2.4: Characteristics of the artificial RNA structures; A) Test set name, number of instances and their sizes, B) General features of all structures of Ia and Ib, minimal and maximal sizes are given.

We generated two test sets of 300 structures each. Test set Ia consists of short structures up to a length of 200, while test set Ib includes larger structures of sizes between 300 and 700. Even if test sets Ia and Ib also include multiloop structures, we additionally analyzed two small but complex ML structures in test set Ic. Structure Ic-1 turned out to be hard to design because it includes a stem just consisting of only one base pair. None of the tested programs managed it to design a sequence folding into this structure. Structure Ic-2 differs from Ic-1 just in the challenging stem, which consists of three base pairs here (see Figure 2.5). Our results show that this slight difference made the structure much easier to design.

Using the artificial test sets Ia, Ib, and Ic, we analyzed success and speed of INFO-RNA, RNA-SSD, and RNAinverse. Please note that due to the large sizes of the structures included in test set Ib the p-mode of RNAinverse was not applied to this test set. For test set Ia, we examined each structure 100 times with each algorithm and counted the structures, for which the respective algorithm succeeded in all 100 cases. The same was done for test set Ib, but here, each structure was examined only ten times.
Table 2.5 summarizes the results. INFO-RNA was always successful for all 300 structures of Ia as well as of Ib. For small structures (Ia), the success rates of

Name	INFO-RNA	RNA-SSD	RNAinverse
Ia (CSR)	300/300	298/300	294/300
Ia ($\bar{E_T}$)	0.1	0.2	41.9
Ib (CSR)	300/300	294/300	1/300
Ib ($\bar{E_T}$)	9.1	46.8	-

Table 2.5: Success and speed of INFO-RNA, RNA-SSD, and RNAinverse when analyzing artificial test sets Ia and Ib. The complete success rate (CSR) gives the fraction of structures for which the respective algorithm found a solution in all runs done for each structure. $\bar{E_T}$ represents the average expected time (over all structures) needed to compute a solution (given in CPU seconds).

RNA-SSD and RNAinverse were only a little worse compared to INFO-RNA. Concerning the runtime, RNA-SSD needed twice as long and RNAinverse 400 times as long as INFO-RNA. For test set Ib, which includes larger structures, RNA-SSD also performed only a little worse in the number of successful runs than INFO-RNA but was much slower.

The structures of Ic were examined 100 times each. The resulting success rates and expected computation times are given in Table 2.6. Since for structure Ic-1 all algorithms failed in all cases, no runtimes are given. But all three algorithms designed sequences, whose mfe structures are in a small distance to the target structure. In Table 2.6, the success rates in parentheses give the fraction of sequences whose mfe structures have a distance of at most two to the target structure. For structure Ic-2, the fraction of successful runs differs among all three algorithms. While INFO-RNA failed in only one run (out of 100), RNA-SSD did not succeed in 38 cases. Furthermore, RNA-SSD is more than the 3000 times slower than INFO-RNA. It can be summarized that, for test set Ic, INFO-RNA has a better success rate than the other two algorithms and is much faster.

Further analyzing structure Ic-1, we found that it is impossible to design in the thermodynamic model. This is proven in the following. We will show that the multiloop not including the one-base-pair stem (see Figure 2.6B) is always favorable concerning the used thermodynamic model.

Name	INFO-RNA	RNA-SSD	RNAinverse
Ic-1 (SR)	0/100	0/100	0/100
Ic-1 (E_T)	-	-	-
Ic-1(1) (SR)	(100/100)	(87/100)	(79/100)
Ic-1(1) (E_T)	(6.1)	(2484)	(9.4)
Ic-2 (SR)	99/100	62/100	44/100
Ic-2 (E_T)	0.6	1996.8	21.34

Table 2.6: Performance and speed of INFO-RNA, RNA-SSD, and RNAinverse when analyzing the artificial structures of test set Ic. The success rate (SR) gives the fraction of runs during which the respective algorithm managed to design a sequence that folds into the target structure. E_T represents the expected time needed to compute a solution (given in CPU seconds). The values in parentheses in lines Ic-1(1) give the success rates and the expected times to design a sequence having a mfe structure within a structure distance of one from Ic-1.

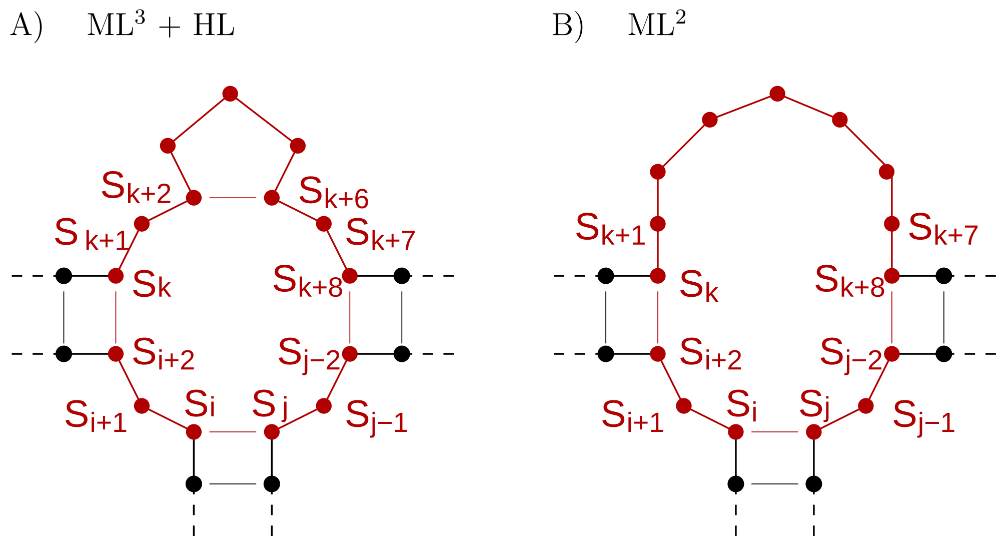


Figure 2.6: A) Undesignable multiloop of structure Ic-1 that contains a one-base-pair stem; B) Favorable multiloop structure instead of A. Relevant structural parts are given in red.

Theorem 2.5.1
Given a secondary structure T, let $\boldsymbol{\mathcal{S}}$ be the set of compatible sequences S as defined in Definition 2.3.2. Furthermore, let $\boldsymbol{ML^i}$ be a multiloop having i included base pairs. Let $\boldsymbol{e(ML^3\ +\ HL, S)}$ be the free energy of the ML and the HL given in red in Figure 2.6A and $\boldsymbol{e(ML^2, S)}$ be the free energy of the ML given in red in Figure 2.6B for a sequence $S \in \mathcal{S}$. Then, it holds

$$\forall S \in \mathcal{S}: \quad e(ML^2, S) < e(ML^3 + HL, S)$$

and thus ML^2 is the favorable multiloop in the structure for any sequence $S \in \mathcal{S}$.

Proof of Theorem 2.5.1

$e(\mathrm{ML}^3\ +\ \mathrm{HL}, S)$ is composed of seven terms:

$$\begin{aligned} e(\mathrm{ML}^3\ +\ \mathrm{HL}, S) \quad = \quad & e_{\mathrm{ML}}(3,4) \\ & + e^{db}(S_{i+1} | (S_{i+2}, S_k), (S_i, S_j)) \\ & + e^{db}(S_{k+1} | (S_{k+2}, S_{k+6}), (S_{i+2}, S_k)) \\ & + e^{db}(S_{k+7} | (S_{k+8}, S_{j-2}), (S_{k+2}, S_{k+6})) \\ & + e^{db}(S_{j-1} | (S_{k+8}, S_{j-2}), (S_i, S_j)) \\ & + e_{\mathrm{HL}}(3) + 2 * e_{\mathrm{AU}}((k+2, k+6), (S_{k+2}, S_{k+6})) \end{aligned} \tag{2.12}$$

where $e^{db}(S_x | (S_{x+1}, S_y), (S_{x-1}, S_z))$ is the dangling base energy of the free base S_x that is adjacent to base pairs (S_{x+1}, S_y) and $(S_{x-1}, S_z))$. It equals the minimum of the dangling base energies concerning both adjacent pairs. That is:

$$e^{db}(S_x | (S_{x+1}, S_y), (S_{x-1}, S_z)) = \min\{e^{db}(S_x | (S_{x+1}, S_y)), e^{db}(S_x | (S_{x-1}, S_z))\} \tag{2.13}$$

Similarly, $e(\mathrm{ML}^2, S)$ is composed of five terms:

$$\begin{aligned} e(\mathrm{ML}^2, S) \quad = \quad & e_{\mathrm{ML}}(2,9) \\ & + e^{db}(S_{i+1} | (S_{i+2}, S_k), (S_i, S_j)) \\ & + e^{db}(S_{k+1} | (S_{i+2}, S_k)) \\ & + e^{db}(S_{k+7} | (S_{k+8}, S_{j-2})) \\ & + e^{db}(S_{j-1} | (S_{k+8}, S_{j-2}), (S_i, S_j)). \end{aligned} \tag{2.14}$$

Now, we show that $e(\text{ML}^3+\text{HL}, S)-e(\text{ML}^2, S) > 0$, which is equivalent to showing $e(\text{ML}^2, S) < e(\text{ML}^3+\text{HL}, S)$.

$$\begin{aligned} & e(\text{ML}^3+\text{HL}, S) - e(\text{ML}^2, S) = \\ & \quad = \; e_{\text{ML}}(3,4) - e_{\text{ML}}(2,9) \\ & \qquad +e^{db}(S_{k+1}|(S_{k+2}, S_{k+6}), (S_{i+2}, S_k)) - e^{db}(S_{k+1}|(S_{i+2}, S_k)) \\ & \qquad +e^{db}(S_{k+7}|(S_{k+8}, S_{j-2}), (S_{k+2}, S_{k+6})) - e^{db}(S_{k+7}|(S_{k+8}, S_{j-2})) \\ & \qquad +e_{\text{HL}}(3) + 2 * e_{\text{AU}}((k+2, k+6), (S_{k+2}, S_{k+6})) \end{aligned}$$

Since $e^{db}(S_x|(S_{x+1}, S_y), (S_{x-1}, S_z))$ is determined by a minimum over two separate dangling base energies as described in Equation 2.13, the above given equation can be restated as

$$\begin{aligned} & e(\text{ML}^3+\text{HL}, S) - e(\text{ML}^2, S) = \\ & = \; a + b*4 + c*(3+1) - (a + b*9 + c*(2+1)) \\ & \quad + \min\{e^{db}(S_{k+1}|(S_{k+2}, S_{k+6})), e^{db}(S_{k+1}|(S_{i+2}, S_k))\} - e^{db}(S_{k+1}|(S_{i+2}, S_k)) \\ & \quad + \min\{e^{db}(S_{k+7}|(S_{k+8}, S_{j-2})), e^{db}(S_{k+7}|(S_{k+2}, S_{k+6}))\} - e^{db}(S_{k+7}|(S_{k+8}, S_{j-2})) \\ & \quad +e_{\text{HL}}(3) + 2 * e_{\text{AU}}((k+2, k+6), (S_{k+2}, S_{k+6})) \\ & \\ & = \; -5*b + c \\ & \quad + \min\{e^{db}(S_{k+1}|(S_{k+2}, S_{k+6})) - e^{db}(S_{k+1}|(S_{i+2}, S_k)), 0\} \\ & \quad + \min\{e^{db}(S_{k+7}|(S_{k+2}, S_{k+6})) - e^{db}(S_{k+7}|(S_{k+8}, S_{j-2})), 0\} \\ & \quad +e_{\text{HL}}(3) + 2 * e_{\text{AU}}((k+2, k+6), (S_{k+2}, S_{k+6})) \end{aligned} \tag{2.15}$$

Let $e^{db}_{down}(S_i)$ and $e^{db}_{up}(S_i)$ be the dangling base energy of a free base S_i adjacent to a base pair in a ML or an EL, which is located downstream and upstream of the adjacent base in the base pair, respectively. According to Mathews *et al.* [MSZT99], dangling base energies can adopt the following values:

$$
\begin{array}{lcl}
e^{db}_{down}(A) & \in & \{-1.7, -1.1, -0.8, -0.7\} \\
e^{db}_{down}(C) & \in & \{-0.8, -0.5, -0.4, -0.1\} \\
e^{db}_{down}(G) & \in & \{-1.7, -1.3, -0.8, -0.7\} \\
e^{db}_{down}(U) & \in & \{-1.2, -0.6, -0.1\} \\
 & & \\
e^{db}_{up}(A) & \in & \{-0.5, -0.3, -0.2\} \\
e^{db}_{up}(C) & \in & \{-0.3, -0.1\} \\
e^{db}_{up}(G) & \in & \{-0.4, -0.2, 0.0\} \\
e^{db}_{up}(U) & \in & \{-0.2, -0.1, 0.0\}
\end{array}
\tag{2.16}
$$

Furthermore, according to Mathews *et al.* [MSZT99], parameters for the ML energy fraction are set to

$$
a = 3.4 \qquad b = 0.0 \qquad c = 0.4. \tag{2.17}
$$

and the energy fraction $e_{\mathrm{HL}}(3)$ of a hairpin loop of size 3 is specified as follows:

$$
e_{\mathrm{HL}}(3) = 5.7
$$

Thus using the best parameters of Equation 2.16 and neglect the consistent assignment of base pair (S_{k+2}, S_{k+6}), we can find a lower bound for Equation 2.15:

$$
\begin{aligned}
& e(\mathrm{ML}^3 + \mathrm{HL}, S) - e(\mathrm{ML}^2, S) \\
& \quad \geq \; 0.4 \\
& \qquad + \min\{ \min_{S_{k+1} \in \sum^B} \{ e^{db}_{up}(S_{k+1}) - e^{db}_{down}(S_{k+1}) \}, 0 \} \\
& \qquad + \min\{ \min_{S_{k+7} \in \sum^B} \{ e^{db}_{down}(S_{k+7}) - e^{db}_{up}(S_{k+7}) \}, 0 \} \\
& \qquad +5.7 + 0 \\
& \quad = \; 6.1 + (-0.3) - (-0.1) + (-1.7) - (0.0) \\
& \quad = \; 4.2
\end{aligned}
\tag{2.18}
$$

Due to the dependencies of dangling base energies of the free base upstream and downstream of the additional base pair (S_{k+2}, S_{k+6}), it is possible that this lower bound cannot be realized. It holds

$$e(\mathrm{ML}^3 + \mathrm{HL}, S) - e(\mathrm{ML}^2, S) \geq 4.2 > 0$$

and hence

$$e(\mathrm{ML}^2, S) < e(\mathrm{ML}^3 + \mathrm{HL}, S).$$

$\square$

We have shown, that the multiloop structure Ic-1 given in Figure 2.5 is not designable when using the given thermodynamic energy model. This holds since the multiloop without the one-base-pair stem is more stable than the multiloop with the additional stem including the hairpin loop for every sequence $S \in \mathcal{S}$. In the following, we can proof an even stronger characteristic of these two alternative structures. That is, even the worst assignment of the multiloop ML^2 is more stable than the best assignment of multiloop ML^3 including the HL.

Theorem 2.5.2
Let S, $\mathcal{S}$, ML^i, $e(ML^3 + HL, S)$, and $e(ML^2, S)$ be defined as in Theorem 2.5.1. Then, it holds

$$\max_{S \in \mathcal{S}} e(ML^2, S) < \min_{S \in \mathcal{S}} e(ML^3 + HL, S).$$

Proof of Theorem 2.5.2
The energy fraction $e(\mathrm{ML}^3 + \mathrm{HL}, S)$ of multiloop ML^3 and the additional hairpin loop and the energy fraction $e(\mathrm{ML}^2, S)$ of multiloops ML^2 are calculated as given in Equations 2.12 and 2.14.

In the following, we show that

$$\min_{S \in \mathcal{S}} e(\mathrm{ML}^3 + \mathrm{HL}, S) - \max_{S \in \mathcal{S}} e(\mathrm{ML}^2, S) > 0$$

which equals the proposition of Theorem 2.5.2.

$$\min_{S\in\mathcal{S}} e(\mathrm{ML}^3+\mathrm{HL},S) - \max_{S\in\mathcal{S}} e(\mathrm{ML}^2,S) =$$

$$\begin{aligned}
&= e_{\mathrm{ML}}(3,4) - e_{\mathrm{ML}}(2,9)\\
&\quad + \min_{S\in\mathcal{S}} e^{db}(S_{k+1}|(S_{k+2},S_{k+6}),(S_{i+2},S_k)) - \max_{S\in\mathcal{S}} e^{db}(S_{k+1}|(S_{i+2},S_k))\\
&\quad + \min_{S\in\mathcal{S}} e^{db}(S_{k+7}|(S_{k+8},S_{j-2}),(S_{k+2},S_{k+6})) - \max_{S\in\mathcal{S}} e^{db}(S_{k+7}|(S_{k+8},S_{j-2}))\\
&\quad + e_{\mathrm{HL}}(3) + \min_{S\in\mathcal{S}} 2 * e_{\mathrm{AU}}((k+2,k+6),(S_{k+2},S_{k+6}))\\
&= 0.4\\
&\quad + \min_{S\in\mathcal{S}}\{\min\{e^{db}_{down}(S_{k+1}), e^{db}_{up}(S_{k+1})\}\} - \max_{S\in\mathcal{S}} e^{db}_{down}(S_{k+1})\\
&\quad + \min_{S\in\mathcal{S}}\{\min\{e^{db}_{down}(S_{k+7}), e^{db}_{up}(S_{k+7})\}\} - \max_{S\in\mathcal{S}} e^{db}_{up}(S_{k+7})\\
&\quad +5.7+0\\
&= 6.1 + (-1.7) - (-0.1) + (-1.7) - 0.0\\
&= 2.8 > 0
\end{aligned}$$

$\square$

Generally, short stems are often not stable enough to compensate for the penalties associated with the adjacent loops [AHHC07]. In our example, the single-base-pair stem is not stable enough to compensate the hairpin loop energy. This is mainly due to the fact that the free base penalty in the current thermodynamic model is set to 0 as given in Equation 2.17.

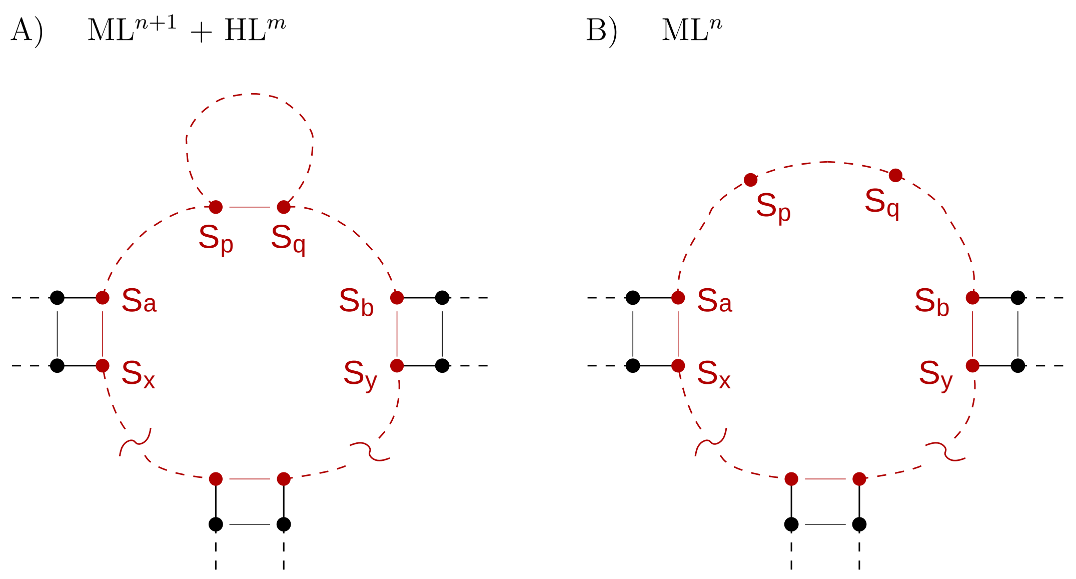


Figure 2.7: A) Undesignable multiloop including a one-base-pair stem; B) Favorable multiloop structure instead of A. Relevant structural parts are given in red, dashed lines indicate variable numbers of free bases, dashed lines that are split up by a $\sim$ represent variable structure parts, i.e. there could be several stems and/or free bases. Furthermore, base pair (S_x, S_a) or base pair (S_b, S_y) could also replace the closing pair of the ML. In some special cases where no or only one free base is between base pair (S_p, S_q) and base pair (S_x, S_a) or base pair (S_b, S_y), S_a can equal S_{p-1} or S_{p-2} and S_b can equal S_{q+1} or S_{q+2}.

Theorem 2.5.3

Given a secondary structure T, let $\mathcal{S}$ be the set of compatible sequences S as defined in Definition 2.3.2. Furthermore, let ML^i be a multiloop having i included base pairs and HL^j be a hairpin loop of size j. Let $e(ML^{n+1} + HL^m, S)$ be the free energy fraction of a multiloop with $n+1$ included pairs, where one of the pairs is an one-base-pair stem that separates a hairpin loop of size m from the multiloop (given in red in Figure 2.7A), and $e(ML^n, S)$ be the free energy of a multiloop with n included pairs given in red in Figure 2.7B for a sequence $S \in \mathcal{S}$. Then, it holds

$$\forall S \in \mathcal{S} : \quad e(ML^n, S) < e(ML^{n+1} + HL^m, S)$$

and thus ML^n is the favorable multiloop in the structure for any sequence $S \in \mathcal{S}$.

Theorem 2.5.3 generalizes Theorem 2.5.1 to the undesignability of any multiloop including a one-base-pair stem. Its proof is given in the following.

Proof of Theorem 2.5.3
Here, we only need to take into account the parts of the multiloop energy fraction that differ between the two cases shown in Figure 2.7. The respective energy parts are named $e^{rel}(.)$. Thus, the relevant energy parameters for the multiloop with $n+1$ included pairs and the additional hairpin loop of m bases are as follows:

$$\begin{aligned} e^{rel}(\mathrm{ML}^{n+1} + \mathrm{HL}^{m}, S) \;=\; & e_{\mathrm{ML}}(n+1, f-m-2) + \\ & + \begin{cases} e^{db}(S_{p-1}|(S_p, S_q), (S_x, S_a)) & \text{if there is only one free base} \\ & \text{between } S_a \text{ and } S_p\text{, i.e.} \\ & a+1 = p-1 \\ e^{db}(S_{p-1}|(S_p, S_q)) + e^{db}(S_{a+1}|(S_x, S_a)) & \text{if there are } > 1 \text{ free bases} \\ & \text{between } S_a \text{ and } S_p \\ 0 & \text{if there is no free base} \\ & \text{between } S_a \text{ and } S_p \end{cases} \\ & + \begin{cases} e^{db}(S_{q+1}|(S_p, S_q), (S_b, S_y)) & \text{if there is only one free base} \\ & \text{between } S_q \text{ and } S_b\text{, i.e.} \\ & b-1 = q+1 \\ e^{db}(S_{q+1}|(S_p, S_q)) + e^{db}(S_{b-1}|(S_b, S_y)) & \text{if there are } > 1 \text{ free bases} \\ & \text{between } S_q \text{ and } S_b \\ 0 & \text{if there is no free base} \\ & \text{between } S_q \text{ and } S_b \end{cases} \\ & + e_{\mathrm{HL}}(m) + 2 * e_{\mathrm{AU}}((p,q), (S_p, S_q)) \end{aligned} \tag{2.19}$$

where f represents the number of free bases in a multiloop as given in Table 2.2.

Relevant energy parts of the multiloop with only n included base pairs are summarized in $e^{rel}(\mathrm{ML}^{n}, S)$. They are independent of the numbers of free bases between S_a and S_p and between S_p and S_b.

$$e^{rel}(\mathrm{ML}^{n}, S) = e_{\mathrm{ML}}(n, f) + e^{db}(S_{a+1}|(S_x, S_a)) + e^{db}(S_{b-1}|(S_b, S_y)) \tag{2.20}$$

In the following, we show that

$$e^{rel}(\mathrm{ML}^{n+1} + \mathrm{HL}^{m}, S) - e^{rel}(\mathrm{ML}^{n}, S) > 0$$

which equals the proposition of Theorem 2.5.3. Using Equations 1.2, 2.19, 2.20, and 2.13, we get the following:

$$\begin{aligned}
& e^{rel}(\mathrm{ML}^{n+1} + \mathrm{HL}^{m}, S) - e^{rel}(\mathrm{ML}^{n}, S) \\
&= a + b*(f-m-2) + c*(n+2) - a - b*f - c*(n+1) \\
&+ \begin{cases}
\min\{e^{db}(S_{p-1}|(S_p,S_q)) - e^{db}(S_{p-1}|(S_x,S_a)), 0\} \\
\quad \text{if there is only one free base between } S_a \text{ and } S_p\text{, i.e. } p-1=a+1 \\
e^{db}(S_{p-1}|(S_p,S_q)) \\
\quad \text{if there are } >1 \text{ free bases between } S_a \text{ and } S_p \\
-e^{db}(S_{a+1}|(S_x,S_a)) \\
\quad \text{if there is no free base between } S_a \text{ and } S_p
\end{cases} \\
&+ \begin{cases}
\min\{e^{db}(S_{q+1}|(S_p,S_q)) - e^{db}(S_{q+1}|(S_b,S_y)), 0\} \\
\quad \text{if there is only one free base between } S_q \text{ and } S_b\text{, i.e. } q+1=b-1 \\
e^{db}(S_{q+1}|(S_p,S_q)) \\
\quad \text{if there are } >1 \text{ free bases between } S_q \text{ and } S_b \\
-e^{db}(S_{b-1}|(S_b,S_y)) \\
\quad \text{if there is no free base between } S_q \text{ and } S_b
\end{cases} \\
&+ e_{\mathrm{HL}}(m) + 2*e_{\mathrm{AU}}((p,q),(S_p,S_q))
\end{aligned}$$

To get a lower bound, this equation can be reduced further. However, due to the dependencies of dangling base energies of the free base upstream and downstream of the additional base pair (S_p, S_q), it is possible that this lower bound cannot be realized, which is no problem in our case. Furthermore, values given in 2.17 are utilized and $e^{db}_{down}(S_i)$ and $e^{db}_{up}(S_i)$ are used as given in Equation 2.16.

$$
\begin{aligned}
& e^{rel}(\mathrm{ML}^{n+1} + \mathrm{HL}^{m}, S) - e^{rel}(\mathrm{ML}^{n}, S) \\
& \quad \geq \; 0.4 + \\
& \quad + \begin{cases}
\min\{\min\limits_{S_{p-1}\in\sum^B} \{e^{db}_{up}(S_{p-1}) - e^{db}_{down}(S_{p-1})\}, 0\} \\
\qquad \text{if there is only one free base between } S_a \text{ and } S_p \\
\min\limits_{S_{p-1}\in\sum^B} e^{db}_{up}(S_{p-1}) \\
\qquad \text{if there are } > 1 \text{ free bases between } S_a \text{ and } S_p \\
-\max\limits_{S_{a+1}\in\sum^B} e^{db}_{down}(S_{a+1}) \\
\qquad \text{if there is no free base between } S_a \text{ and } S_p
\end{cases} \\
& \quad + \begin{cases}
\min\{\min\limits_{S_{q+1}\in\sum^B} \{e^{db}_{down}(S_{q+1}) - e^{db}_{up}(S_{q+1})\}, 0\} \\
\qquad \text{if there is only one free base between } S_q \text{ and } S_b \\
\min\limits_{S_{q+1}\in\sum^B} e^{db}_{down}(S_{q+1}) \\
\qquad \text{if there are } > 1 \text{ free bases between } S_q \text{ and } S_b \\
-\max\limits_{S_{b-1}\in\sum^B} e^{db}_{up}(S_{b-1}) \\
\qquad \text{if there is no free base between } S_q \text{ and } S_b
\end{cases} \\
& \quad + e_{\mathrm{HL}}(m) + 2 * \min\limits_{(S_p,S_q)\in\sum^{BP}} e_{\mathrm{AU}}((p,q),(S_p,S_q))
\end{aligned}
$$

To get the minimal distance of $e^{rel}(\text{ML}^{n+1} + \text{HL}^{m}, S)$ and $e^{rel}(\text{ML}^{n}, S)$, the case that gives the smallest value in each case discrimination has to be chosen.

$$\begin{aligned}
& e^{rel}(\text{ML}^{n+1} + \text{HL}^{m}, S) - e^{rel}(\text{ML}^{n}, S) \\
& \geq \quad 0.4 + \min \left\{ \begin{array}{l} -0.3 - (-0.1) \\ -0.5 \\ -(-0.1) \end{array} \right\} + \min \left\{ \begin{array}{l} -1.7 - 0.0 \\ -1.7 \\ -0.0 \end{array} \right\} + e_{\text{HL}}(m) + 0 \\
& = \quad 0.4 + (-0.5) + (-1.7) + e_{\text{HL}}(m) \\
& = \quad -1.8 + e_{\text{HL}}(m) \\
& > 0
\end{aligned} \tag{2.21}$$

The last step in Equation 2.21 holds since

$$\forall m \geq 3 : e_{\text{HL}}(m) > 2.6$$

□

2.5.2 Biological Test Sets

Computationally predicted structures for known RNA sequences. In order to test the performance for real biological data, we used two further test sets. These sets consist of structures that are computationally predicted for known RNA sequences. All structures were predicted by RNAfold from the Vienna RNA Package [HFS+94], the same procedure that is used to evaluate the foldings during INFO-RNA, RNA-SSD, and RNAinverse. Thus, it is guaranteed that at least one sequence exists, whose mfe structure is the one to be designed.

The first test set of computationally predicted structures consists of 24 structures of 260-1475 bases also analyzed by Andronescu *et al.* [AFH+04]. They created a set of ribosomal RNA sequences obtained from the Ribosomal Database Project [CCM+03] and predicted their mfe structures. We refer to this as test set IIa. Since Andronescu *et al.* have already shown that RNA-SSD performs better than RNAinverse when analyzing structures of IIa, we restricted our tests to a comparison of INFO-RNA and RNA-SSD.

For each structure, we performed between ten and 50 runs per algorithm similar to Andronescu *et al.* As both algorithms are successful here, we turned our attention to the comparison of speed. For that purpose, we applied INFO-RNA in a slightly different way. If it did not succeed within 300 CPU seconds, INFO-RNA is aborted and, thus, terminated unsuccessfully. Afterwards, the neighbor-testing-mode is changed for the next runs (from the energy-dependent NE-mode to the arbitrary NA-mode or back), as it seems that the current strategy is not successful for the given structure. The new mode is retained until it fails. Thus, we always applied the mode with less unsuccessful terminations. If both modes led to the same number of failures, NE-mode was chosen. This strategy of testing is obvious, since users of the program will change the parameters as well, if the algorithm has failed with their current parameter values. Since RNA-SSD does not include these modes, the strategy was only applied in case of INFO-RNA. Evaluating test set IIa, the results are comparable for INFO-RNA and RNA-SSD. However, INFO-RNA failed for only one structure, while RNA-SSD did for three. Detailed results are given in Table 2.7.

The second test set of computationally predicted structures includes 308 structures with a length between 220 and 1975 bases. They are the minimum free energy structures of all annotated eukaryotic rRNA gene sequences from release 9.27 of the Ribosomal Database Project (RDP-II) [CCF$^+$05]. We refer to this as test set IIb. The whole set of eukaryotic sequences from RDP-II was chosen since the performance of INFO-RNA has to be tested on more than some exemplary sequences chosen by Andronescu *et al.* For INFO-RNA and RNA-SSD, we performed 25 runs for each structure, and because of the longer runtime only ten runs for RNAinverse. Furthermore, runs were terminated unsuccessfully if no solution was found after 3600 CPU seconds. To determine the success of the algorithms for classes of structures in a certain size range, we divided test set IIb into three subsets according to the size of the structures. The results are shown in Table 2.8. INFO-RNA performs best and fastest for all three subsets of IIb. Additionally, all runs for each structure were successful.

Structures from the biological literature. Finally, we analyzed the performance of INFO-RNA and RNA-SSD on a test set containing structures published in the literature. This set is identical to the test set C in Andronescu *et al.* We refer to it as test set III. Pseudoknots were removed by disregarding pairs in pseudoknots. Results are given in Table 2.9. We did not examine the performance of RNAinverse since Andronescu *et al.* have already done this. To analyze success and speed of INFO-RNA and RNA-SSD, we examined 100 runs per structure for each algo-

	Name (GenBank [BKML+07] accession number)	Size	INFO-RNA E_T	RNA-SSD E_T
1	Rhizobiaceae group bacterium NR64 (Z83250)	260	1.0	1.9
2	Bacterial sp. from marine plankton (L11935)	264	1.2	1.0
3	Leptospira interrogans strain 94-7997013 (LIU92530)	289	1.7	871.3*
4	Unidentified marine eubacterium (U84629)	299	2.8	40.0
5	Uncultured bacterium SY2-21 (AF107506)	337	2.2	6.0
6	Ochrobactrum sp. BL200-8 (AF106618)	350	1.0	1.0
7	Uncultured eubacterium 3-25 (AJ011149)	376	1.4	43.5
8	Prevotella ruminicola, M384 (S70838)	389	2.9	6.4
9	Uncultured crenarchaeote clone pSL1 (U63350)	418	4.2	2.6
10	Uncult. eubacterium clone CRE-FL72 (AF141485)	473	4.5	32.2
11	Unidentified eubacterium clone vadinIA59 (U81771)	491	2.7	3.3
12	Stenotrophomonas sp., isolate P-26-14 (AJ130779)	506	3.8	3.4
13	Nitrobacter sp. Nb4 (AF096836)	646	5.6	7.8
14	Wolbachia pipientis (X61771)	659	11.5	113.8
15	Uncultured archaeon ST1-4 (AJ236455)	751	273.6*	381.7*
16	Bradyrhizobium sp., isolate 283A (AJ132572)	780	76.5	14.5
17	Spirochaeta sp., clone Hs33 (AB015827)	856	16.6	10.5
18	Sulfolobus acidocaldarius (D38777)	858	121.5	47.5
19	Unidentified methanogen ARC21 (AF029195)	1053	54.2	13.1
20	Planctomyces brasiliensis, DSM 5305 (X81949)	1200	73.7	68.6
21	Uncultured archaeon 'KTK 9A' (AJ133622)	1296	135.0	43.4
22	Methanococcus fervens (AF056938)	1398	72.5	112.9
23	Pseudomonas sp. Y1000 (X99676)	1442	143.1	141.5
24	Methanococcus jannaschii (L77117 b. 157084-159459)	1475	149.8	407.9*

Table 2.7: Results for INFO-RNA and RNA-SSD on test set IIa. Both algorithms were used 50 times to design sequences that fold into structures 1-8, 25 times to design structures 9-16, and ten times to design structures 17-24. The time E_T is the expected time needed to compute a solution. If an algorithm does not compute the correct solution in all runs, times are marked with *.

Subset of IIb	INFO-RNA	RNA-SSD	RNAinverse
220-400 (ASR)	100%	93%	2.0%
220-400 ($\bar{E_T}$)	2.4	226.8	-
400-900 (ASR)	100%	93%	0.3%
400-900 ($\bar{E_T}$)	93.3	285.3	-
900-1975 (ASR)	100%	81%	0.0%
900-1975 ($\bar{E_T}$)	1447.4	3043.9	-

Table 2.8: Results on test set IIb. This set was split into subsets depending on the structure size. We run INFO-RNA and RNA-SSD 25 times each and RNAinverse ten times for each structure. The time $\bar{E_T}$ is the average expected time needed to compute a solution for a structure of the respective subset. The average success rate (ASR) gives the average fraction of successful runs.

rithm. The success rates and expected computing times demonstrate the excellent performance of INFO-RNA. In all but one case, it was faster than RNA-SSD. Furthermore, it succeeded for more structures than RNA-SSD and unsuccessful runs terminated with better approximate solutions.

2.5.3 Test Sets Including Sequence Constraints

Given an RNA secondary structure and constraints on the sequence, INFO-RNA finds an RNA sequence that is going to adopt this structure and to satisfy the constraints. Moreover, violations of the constraints at some positions may be allowed, which can be advantageous in complicated cases.
This extension of INFO-RNA allows the design of cis-acting mRNA elements such as the iron responsive element (IRE) and the polyadenylation inhibition element (PIE). Both elements have conserved sequence positions in loops. By providing binding sites for regulatory proteins, they determine mRNA stability and translation efficiency. The IRE is essential for the expression of proteins that are involved in the iron metabolism [HK96]. It consists of a stem-loop structure. The nucleotides in the hairpin loop as well as the bulged nucleotides were found to be essential for binding of iron-regulatory proteins. Furthermore, the PIE contains two binding sites for U1A proteins [VGM$^+$00]. It consists of a stem structure with two asymmetric internal loops that serve as U1A binding sites. U1A binding

			INFO-RNA		RNA-SSD	
	Name	Size	SR	E_T	SR	E_T
1	Minimal catalytic domains of the hairpin ribozyme satellite RNA of the tobacco ringspot virus (Figure 1(a) in [Fed00])	65	100/100	0.03	100/100	0.04
2	U3 snoRNA 5'-domain from *Chlamydomonas reinhardtii* (Figure 6(B) in [AMK$^+$00])	79	100/100	0.01	100/100	0.02
3	*H. marismortui* 5S rRNA (Figure 2 in [SBEB02])	122	(100/100)(1)	(45.2)	(100/100)(1)	(2163.9)
4	VS Ribozyme from *Neurospora* mitochondria (Figure 1(A) in [LNL01])	167	100/100	0.1	100/100	0.3
5	R180 ribozyme (Figure 2(B) in [SCGZ02])	180	37/100 (63/100)(1)	194.0	58/100 (20/100)(1)	2267.8
6	XS1 ribozyme, *Bacillus subtilis* P RNA-based ribozyme (Figure 2(A) in [MP99])*	314	100/100	19.0	100/100	22.4
7	Homo Sapiens RNase P RNA (Figure 4 in [PGPZP98])*	340	100/100	66.8	94/100	491.1
8	S20 mRNA from *E.coli* (Figure 2 in [Mac92])	372	100/100	110.8	87/100	728.2
9	*Halobacterium cutirubrum* RNAse P RNA (Figure 2 in [HAV$^+$96])*	376	(100/100)(2)	(5026.8)	(1/100)(5)	(220530.0)
10	Group II intron ribozyme D135 from ai5γ (Figure 5 in [SDSP01])	583	100/100	7.9	100/100	3.9

Table 2.9: Results for test set III. Originally pseudoknotted structures are marked with an asterisk (*). Here, pseudoknots are removed by disregarding eight base pairs in each case. All other structures are pseudoknot-free. The success rate (SR) gives the fraction of runs in which the respective algorithm found a correct solution. E_T represents the expected time needed to compute a solution. For structures where no correct solution was found, SR and E_T are given in parentheses. They reflect the fraction in which the best approximate solution was found and the time needed for it, respectively. The distance to the target structure is also given next to the bracketed success rates.

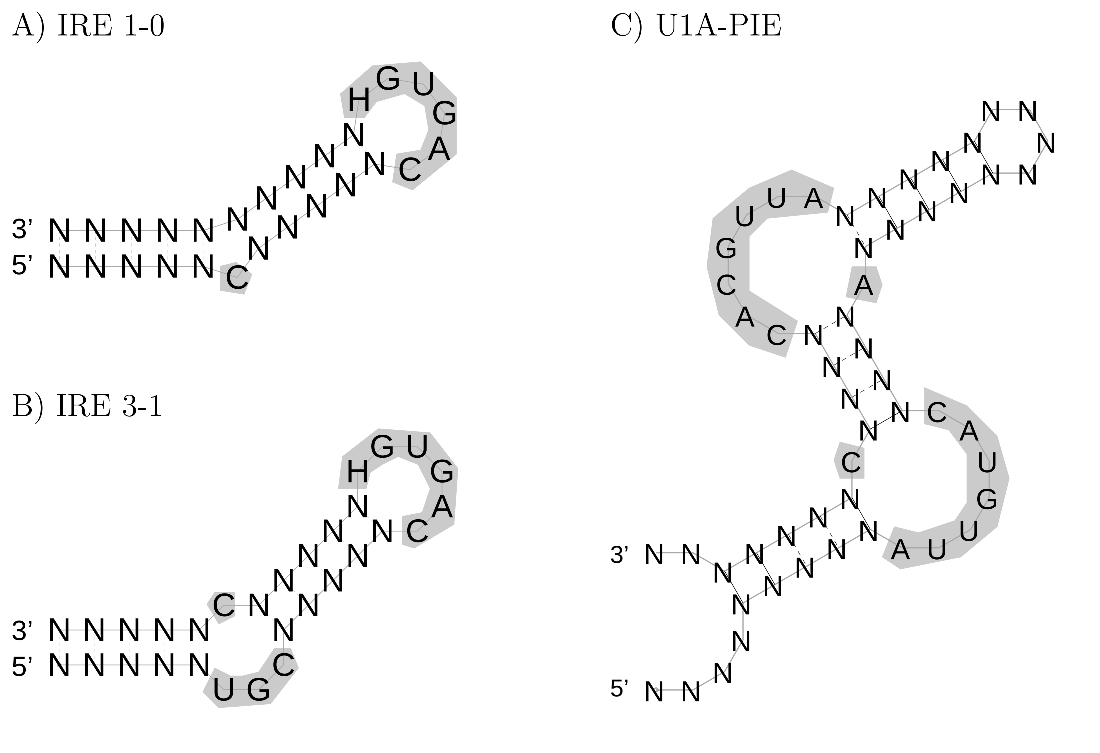


Figure 2.8: The figure shows the consensus structure and conserved sequence positions of (A) an IRE with a single C bulge loop (named IRE 1-0); (B) an IRE with an interior loop of left size 3 and right size 1 (named IRE 3-1); and (C) a PIE that contains two asymmetrical internal loops as binding sites for U1A proteins (called U1A-PIE). Conserved sequence positions are highlighted in gray.

leads to an inhibition of the poly(A) polymerase and a reduced mRNA stability and translation efficiency due to a shortened poly(A) tail. Using INFO-RNA, we designed artificial IREs and PIEs (see Figure 2.8) having a much higher folding probability compared to natural elements (see Figure 2.9). Besides, IREs designed by INFO-RNA adopt the wanted structure as its mfe structure whereas only a small fraction of the natural ones does (see Figure 2.10).

Additionally, we demonstrated the usability of INFO-RNA by designing artificial microRNA (miRNA) precursors that are as stable as possible. To this end, artificial miRNA sequences published in [SOR^{+}06] were used. Applying INFO-RNA, we designed precursors of these artificial miRNAs as well as of the natural miRNA that have a much lower free energy and a higher probability of folding into the target miRNA precursor structure than the natural precursor sequences (see Table 2.10).

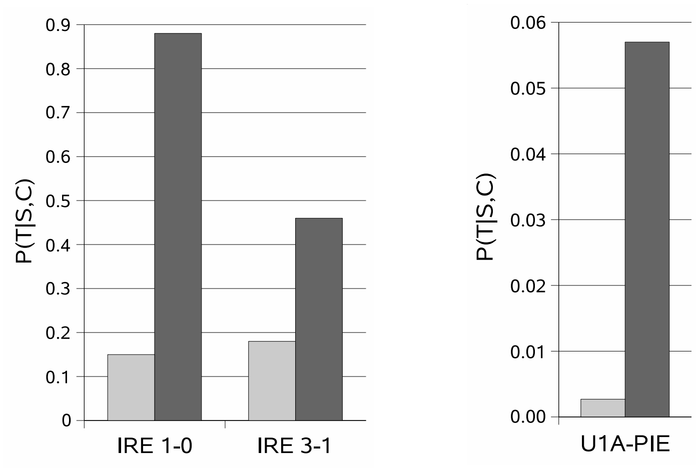


Figure 2.9: The figure shows the folding probabilities $P(T|S,C)$ that the designed IRE and PIE sequences S (fulfilling constraints C) fold into the target structure T. The names of the designed elements are chosen analogously to Figure 2.8. Values for the natural sequences are given in light gray, values for the sequences designed by INFO-RNA in dark gray.

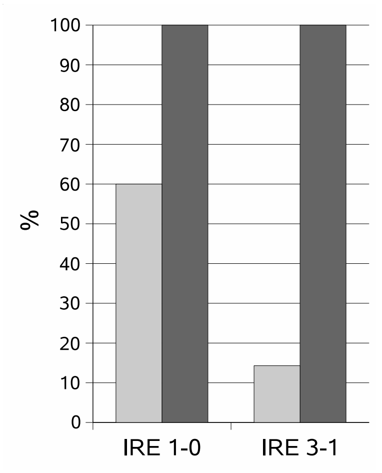


Figure 2.10: The figure shows the fraction of IRE sequences that have the target structure as mfe structure. The names of the designed elements are chosen analogously to Figure 2.8. Values for the natural sequences are given in light gray, values for the sequences designed by INFO-RNA in dark gray.

precursor of artificial miRNAs	minimum free energy [a]	folding probability p [a]	INFO-RNA minimum free energy [b]	INFO-RNA folding probability p^I [b]	$\frac{p^I}{p}$ [c]
pre-amiR-lfy-1	-72.69	0.0246	-160.29	0.1121	4.56
pre-amiR-lfy-2	-72.69	0.0212	-161.49	0.0822	3.88
pre-amiR-white-1	-75.19	0.0266	-161.79	0.0830	3.12
pre-amiR-white-2	-69.29	0.0253	-157.69	0.1462	5.78
pre-amiR-ft-1	-75.19	0.0251	-163.19	0.1938	7.72
pre-amiR-ft-2	-71.49	0.0232	-160.09	0.1306	5.63
pre-amiR-trichome	-75.49	0.0259	-167.09	0.1419	5.48
pre-amiR-mads-1	-69.69	0.0219	-159.99	0.1891	8.63
pre-amiR-mads-2	-72.19	0.0249	-152.59	0.1017	4.08
pre-amiR-yabby-1	-73.49	0.0255	-163.99	0.1531	6.00
pre-amiR-yabby-2	-76.79	0.0272	-157.19	0.1405	5.17
pre-miRNA [d]	-74.49	0.0271	-163.69	0.1252	4.62

[a] values refer to the engineering of the amiRNAs into the natural precursor stem sequence
[b] values refer to using INFO-RNA to design new precursor stem sequences including the amiRNAs
[c] the ratio of the folding probability of sequences designed by INFO-RNA and natural precursor sequences including the amiRNAs
[d] natural miRNA sequence of miR319a

Table 2.10: Minimum free energies and folding probabilities of amiRNA precursors

	IIa		
No. of structure	1	12	17
Size	260	506	856
Best $P(T\|S)$	0.16865	0.01470	$1.46 * 10^{-5}$
Median $P(T\|S)$	0.00901	0.00052	$4.65 * 10^{-8}$
Worst $P(T\|S)$	$4.99 * 10^{-6}$	$9.17 * 10^{-7}$	$2.71 * 10^{-13}$
Biol. $P(T\|S^b)$	0.00023	$4.03 * 10^{-8}$	$1.00 * 10^{-16}$

Table 2.11: Stability of some exemplary results of test sets IIa (chosen according to Andronescu *et al.*). Best $P(T|S)$ gives the highest probability reached by one of our designed sequences for structure T. Median $P(T|S)$ and worst $P(T|S)$ are defined analogously. Biol. $P(T|S^b)$ indicates the folding probability of the biological sequences.

2.5.4 Stability Tests

Generally, the question of the stability of the designed sequences is another important item for the validation of INFO-RNA. To this end, we analyzed the probability of folding into the target structure of some arbitrarily chosen structures of test sets IIa and IIb. The selected biological sequences underlying test set IIa were chosen according to Andronescu *et al.* [AFH+04] to assure comparability. For each, we compared the stability of its mfe fold to that of the designed sequences when folding into the predicted structure. For that purpose, we used the partion function option of RNAfold of the Vienna RNA Package [HFS+94]. For each designed sequence S as well as for the biological sequences S^b, we computed the probability $P(T|S)$ of the final sequence folding into the target structure T. The designed sequences were sorted according to their stability. The best, the median, and the worst ones as well as the results for the biological sequences are given in Table 2.11. Generally, sequences designed by INFO-RNA are much more stable than the biological ones and the sequences obtained by Andronescu *et al.* in [AFH+04]. For the future, it is desirable to design sequences with a stability that is comparable to the stability of the naturally occurring sequences since flexibility is also important in some cases.

In a second step, we analyzed arbitrarily chosen sequences underlying test set IIb (a small, a medium, and a long one) and evaluated the stability of their mfe folds to that of the designed sequences when folding into the predicted structure.

		IIb	
No. of structure	113	258	130
Size	277	716	1225
Best $P(T\|S)$	0.13196	0.00112	$2.46 * 10^{-10}$
Median $P(T\|S)$	0.02637	$2.33 * 10^{-6}$	$2.83 * 10^{-14}$
Worst $P(T\|S)$	0.00059	$6.75 * 10^{-10}$	$3.09 * 10^{-18}$
Biol. $P(T\|S^b)$	$3.60 * 10^{-7}$	$1.67 * 10^{-14}$	$1.86 * 10^{-23}$

Table 2.12: Stability of arbitrarily chosen results of test sets IIb. Best $P(T|S)$ gives the highest probability reached by one of our designed sequences for structure T. Median $P(T|S)$ and worst $P(T|S)$ are defined analogously. Biol. $P(T|S^b)$ indicates the folding probability of the biological sequences.

	IIb (all)	
	$> P(T\|S^b)$	$> 10^3 * P(T\|S^b)$
Best $P(T\|S)$	100%	99%
Median $P(T\|S)$	100%	78.2%
Worst $P(T\|S)$	76.3%	34.7%

Table 2.13: Overall stability results for test set IIb. Values in the table give the fraction of structures T in test set IIb, whose probabilities of the designed sequences (best, median, and worst, respectively) are higher than the probabilities of the natural sequences when folded into T. Terms are chosen analogously to Table 2.12

Results are given in Table 2.12. Again, sequences designed by INFO-RNA have a much higher stability than the biological ones. Furthermore, Table 2.13 shows the high quality of all results of test set IIb. In all cases, the best and even the median designed sequences have a higher stability than the biological ones. For most structures, the best designed sequence was more than 1000 times more stable than the biological one.
All these results show that in INFO-RNA, it is not necessary to optimize the stability additionally. It suffices to minimize the structure distance of the mfe structures to the target one (and the folding probability in some cases) to design highly stable structures.

2.6 Discussion

We have introduced a fast and successful new approach to the inverse RNA folding problem, called INFO-RNA. In general, it outperforms existing tools. It consists of two major steps: a new initialization method and a subsequent advanced stochastic local search that uses an effective neighbor selection method. The former is implemented by a dynamic programming approach, which finds a sequence that among all sequences adopts the target structure with the lowest possible energy. It is done in linear time depending on the structure size. We have shown that this initial sequence is an excellent starting point for the subsequent local search. During the latter, only few local search steps and less time are needed to generate a good sequence that folds into the target structure. This is due to two reasons. Firstly, a generation of a nearly optimal initializing sequence in linear time and secondly, a powerful energy-based pre-ordering of the set of neighbored sequences during the local search, which can be calculated much faster than the actual optimization criteria of minimizing the structure distance or maximizing the folding probability.

To test the performance of INFO-RNA, we analyzed several test sets of artificially generated as well as biological RNA structures and compared the results to RNA-SSD and RNAinverse. In general, INFO-RNA outperforms RNA-SSD and performs substantially better than RNAinverse. However, it should be noted that RNAinverse was also designed to produce random samples from the sequence space. Obviously, INFO-RNA cannot be used to produce samples since the initializing sequence is rather fixed apart from a random variation in unbound bases located in loop regions. We performed some initial experiments to investigate the performance of INFO-RNA using a random initializing sequence. The improved stochastic local search alone is still able to produce comparable results, although INFO-RNA looses much of its speed. Thus for the sampling task, INFO-RNA should be used without the initialization method.

During the local search, we use the fold functions of the Vienna RNA Package, which implements the commonly used RNA structure prediction procedure, the algorithm of Zuker [ZS81]. Thus, INFO-RNA has the same limitations as Zuker's algorithm has: an approximated energy model and the restriction to pseudoknot-free structures. Therefore, sequences designed by INFO-RNA are not guaranteed to fold into the target structure in a cell. But similar to RNA-SSD and RNAinverse, INFO-RNA uses Zuker's algorithm as a subroutine, which can be replaced by another structure prediction algorithm.

When using INFO-RNA in p-mode, it has the same weakness as RNAinverse used in p-mode. It performs slowly and thus cannot be used for larger structures. This problem might be solved when using folding routines as utilized in RNAplfold [BHS06]. RNAplfold evaluates average equilibrium probabilities for all short-range base pairs (i, j) over all fixed-size sequence windows that include i and j. This approach is based on the finding that long-range base pairs are disfavored kinetically relative to short-range pairs [BHS06]. When incorporating the idea of RNAplfold in INFO-RNA, the window size has to be fixed to the distance of the bases of the longest-range pair in the target structure.

Due to the fact that G-C base pairs are energetically most favorable, the initialization sequences of INFO-RNA have a high GC content in general. This GC content is subsequently reduced by the local search but the final sequences are still enriched in G's and C's, which might explain the high stability of the designed sequences. In future, it would be desirable to introduce special constraints in INFO-RNA to reduce the GC content.

To conclude, INFO-RNA is a very fast and successful algorithm for the inverse RNA folding problem, which outperforms existing tools for the most structures.

Chapter 3

Multi Criterial Design of RNA

3.1 Additional Constraints for Functional RNAs

In the previous chapter, we have introduced the inverse RNA folding problem. We looked for non-coding RNA sequences that fold into a given structure and fulfill given constraints on the nucleotide sequence level. In the following, we will deal with an extension of the inverse RNA folding problem. We will additionally take the amino acid sequence into account that is encoded by the designed *messenger RNA (mRNA).* This is needed if one wants to design an RNA sequence that (i) folds into a given structure and (ii) codes for a particular protein. The most prominent example is the translation of an mRNA that codes for a bacterial *selenoprotein* since, here, a special secondary structure is required within the coding part of the mRNA. This example will be discussed in more detail in this chapter.

Generally, an mRNA sequence is translated to an amino acid sequence, which forms the protein. In each case, three consecutive nucleotides specify an amino acid. Such three consecutive nucleotides are called *codon.*

Definition 3.1.1 (Codon)
A **codon** *is a sequence of three consecutive nucleotides* $L_i = S_{3i-2}S_{3i-1}S_{3i}$*, where* $L_i \in (\sum^B)^3 = \{A, C, G, U\}^3$ *is the i-th codon in sequence S.*

The translation of an mRNA into the associated amino acid sequence is determined by the *genetic code*, which is unique for most living organisms [Cri68]. It specifies which codon codes for which amino acid. Generally, each codon codes for a fixed amino acid, while an amino acid is typically encoded by several codons. The genetic code is shown in Table 3.1. For each codon, the translated amino acid is given in one-letter designation as well as in three-letter designation.

	2nd base				
1st base	U	C	A	G	3rd base
U	F Phe	S Ser	Y Tyr	C Cys	U
	F Phe	S Ser	Y Tyr	C Cys	C
	L Leu	S Ser	Stop	Stop	A
	L Leu	S Ser	Stop	W Trp	G
C	L Leu	P Pro	H His	R Arg	U
	L Leu	P Pro	H His	R Arg	C
	L Leu	P Pro	Q Gln	R Arg	A
	L Leu	P Pro	Q Gln	R Arg	G
A	I Ile	T Thr	N Asn	S Ser	U
	I Ile	T Thr	N Asn	S Ser	C
	I Ile	T Thr	K Lys	R Arg	A
	M Met	T Thr	K Lys	R Arg	G
G	V Val	A Ala	D Asp	G Gly	U
	V Val	A Ala	D Asp	G Gly	C
	V Val	A Ala	E Glu	G Gly	A
	V Val	A Ala	E Glu	G Gly	G

Table 3.1: The genetic code.

3.2 Selenoproteins and Selenocysteine Insertion

Selenocysteine is a rare amino acid, which was discovered as the 21st amino acid present in functional proteins. Its three-letter and its one-letter symbol is *Sec* and *U*, respectively. Proteins containing one or more selenocysteines are called *selenoproteins*. They have gained much interest recently since selenoproteins are of fundamental importance to human health. They are an essential component of several major metabolic pathways, including the antioxidant defense systems, the thyroid hormone metabolism, and the immune function (for overview see e.g. [BA01]). Few years ago, the complete mammalian selenoproteome was determined [KCN$^+$03]. There is an enormous interest in the catalytic properties of selenoproteins, especially since selenocysteine is more reactive than its counterpart cysteine. Studies have shown that a selenocysteine containing protein has greatly enhanced enzymatic activities compared to the cysteine homologues [HCF$^+$00].

Selenocysteine is encoded by the UGA-codon, which is usually a STOP-codon (see Table 3.1). It is not included in the standard genetic code shown in Table 3.1 yet. It has been shown that, in the case of selenocysteine, termination of translation is inhibited in the presence of a specific mRNA sequence in the downstream region after the UGA-codon that forms a hairpin-like structure (called **SE**leno**C**ysteine **I**nsertion **S**equence (SECIS), shown in Figure 3.1) [BFHB91]. With the assistance of the special elongation factor SELB [BHB93], the SECIS-element forms a complex consisting of SELB, guanosine-5'-triphosphate (GTP), which is used as a source of energy for the protein synthesis [FLB89], and a specific selenocysteine-tRNA (tRNA$^{\text{Sec}}$) [LZMBB88] (see again Figure 3.1). It is believed that the simultaneous binding of SELB to tRNA$^{\text{Sec}}$ and to the SECIS-element is responsible for the incorporation of selenocysteine into a protein [HWB96].

However, there are differences between the mechanisms for inserting selenocysteine in eukaryotes and bacteria. In eukaryotes, the SECIS-element is located in the 3' *untranslated region (UTR)* of the mRNA with a distance from the UGA-codon that varies from 500 to 5300 nucleotides [LB96] (see Figure 3.2C). In bacteria, the situation is quite different. We consider the case of *E.coli*, where the mechanism of selenocysteine insertion is well understood and all corresponding factors are identified [SHZB91]. The SECIS-element is located immediately downstream the UGA-codon [ZHB90], which implies that the SECIS-element is in the coding part of the protein (see Figure 3.2B). A displacement of the SECIS-element by more than one codon results in a drastic reduction of selenocysteine insertion efficiency [LRGEK98].

To investigate the properties (e.g. structure and function) of selenoproteins, one needs large amounts of pure proteins. Generally, the production of pure proteins is

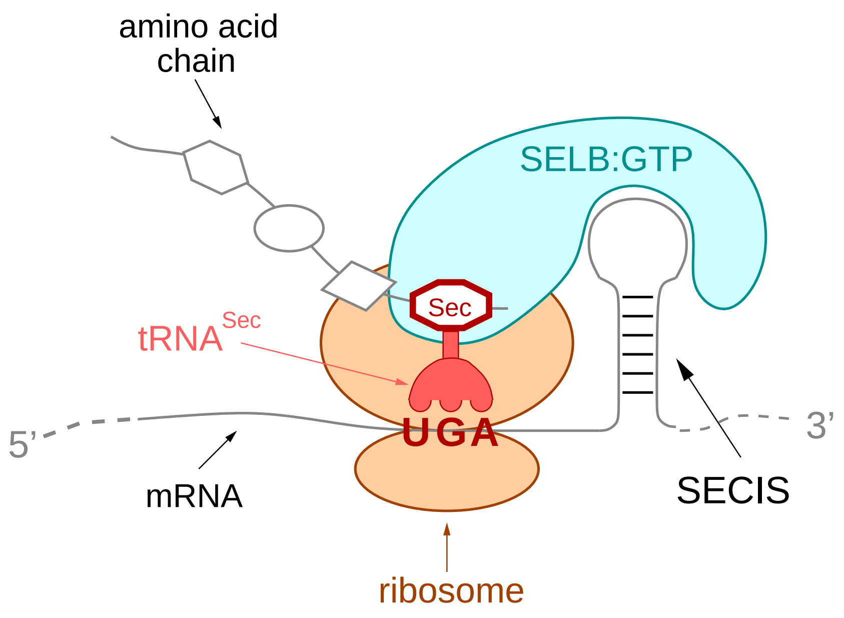


Figure 3.1: Translation of mRNA requires a SECIS-element in case of selenocysteine. Furthermore, the elongation factor SELB, GTP, and a tRNA^{Sec} are needed.

done by using a recombinant protein expression systems with *E.coli* being the simplest system to handle. But especially for eukaryotic selenoproteins, recombinant expression in *E.coli* is complicated and fails very often [THB00]. This is due to the different mechanisms of incorporating selenocysteine in *E.coli* and in eukaryotes. The eukaryotic SECIS-element is located in the 3' UTR and thus missing directly after the UGA codon, where *E.coli* needs it. Therefore, there are only few cases of successive heterologous expression [HCF$^+$00, ASL$^+$99, BNM02], which all required a careful, hand-crafted design of the nucleotide sequence that codes for the protein and additionally forms a SECIS-element following the UGA. Thus, we have the following implications:

1. an eukaryotic selenoprotein cannot directly be expressed in the *E.coli* system,
2. expression of an eukaryotic selenoprotein requires the design of an appropriate SECIS-element directly after the UGA-position (see Figure 3.2B), and
3. this design of a new SECIS-element may change the protein sequence (see Figure 3.3).

Therefore, one has to make a compromise between changes in the protein sequence

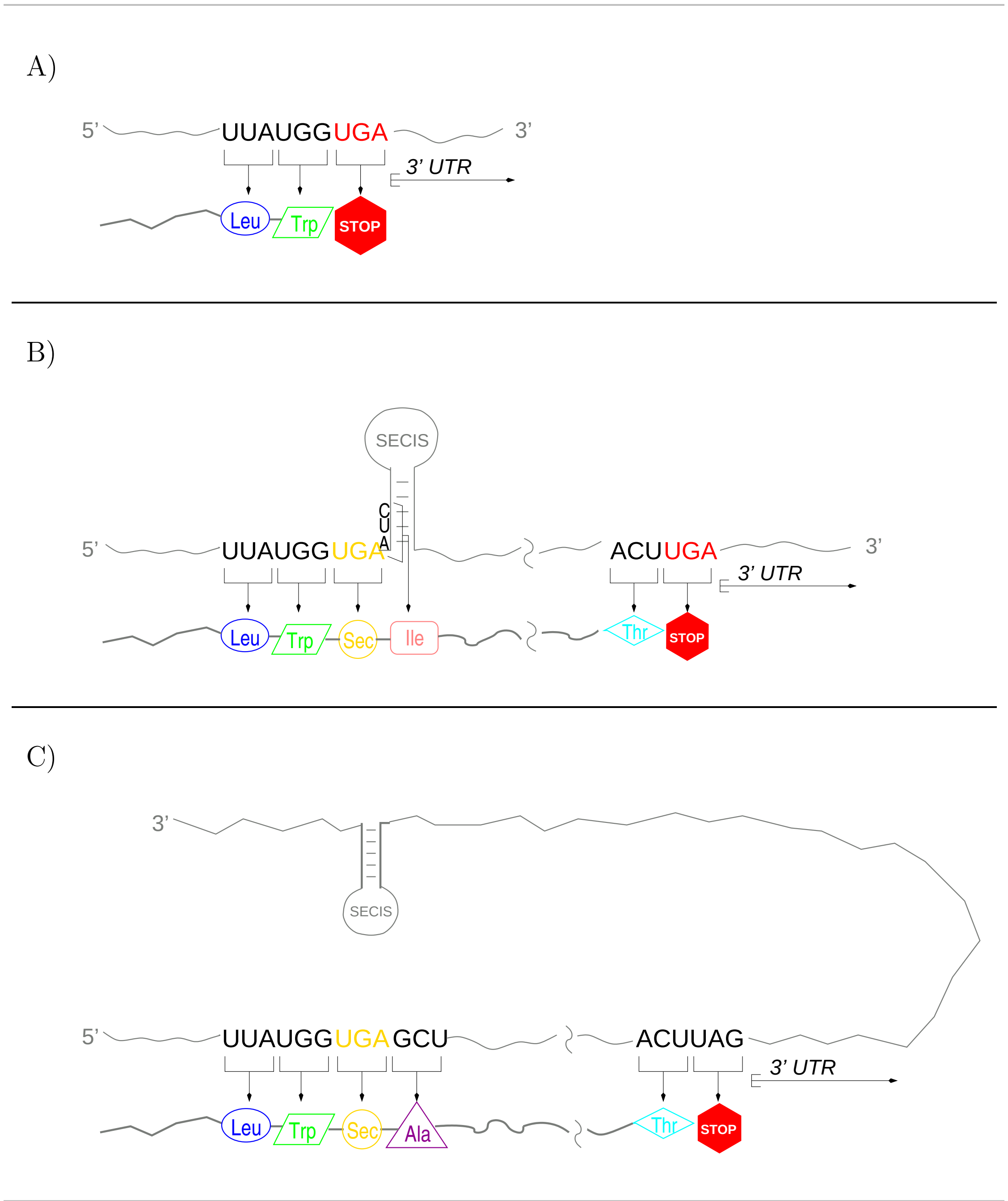
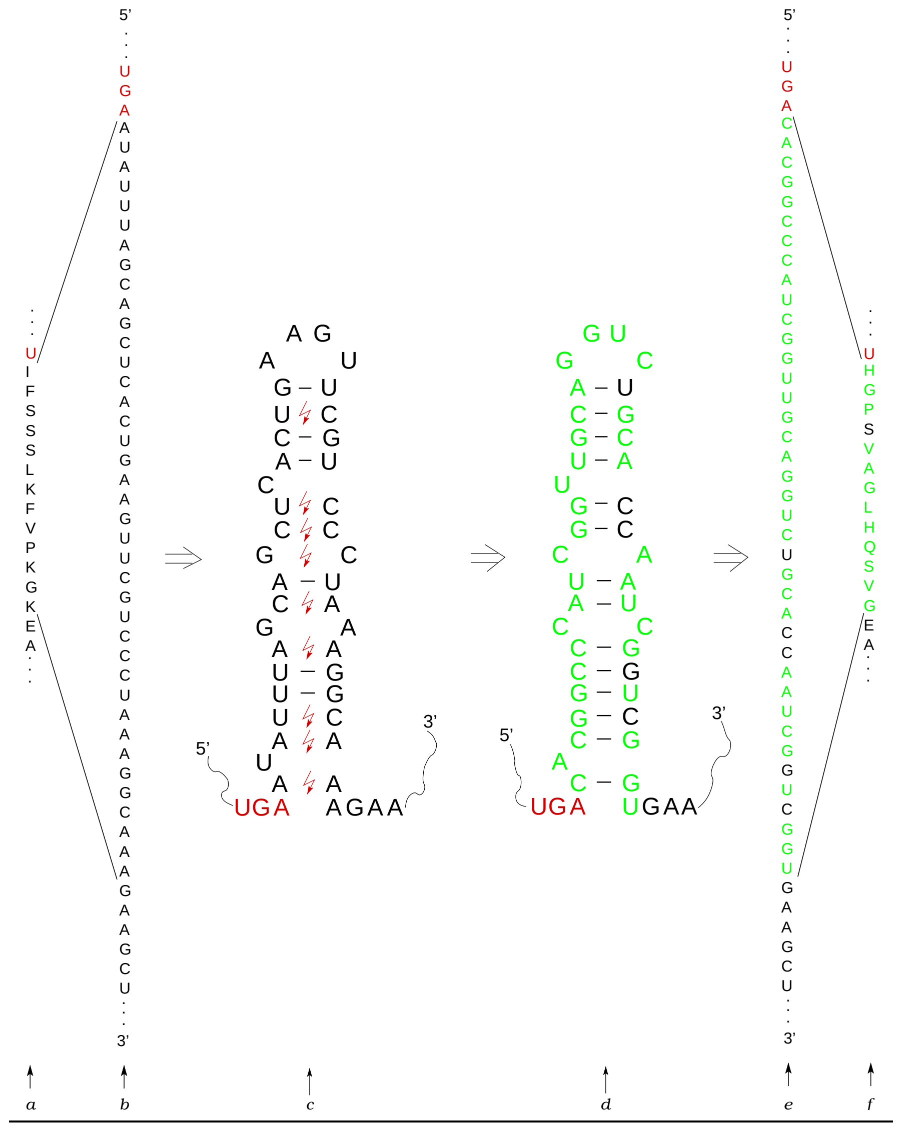


Figure 3.2: A) Function of UGA as stop-codon without any SECIS-element, B) UGA coding for a selenocysteine in *E.coli* having a SECIS-element right after it as well as the function of UGA as a stop-codon without having a consecutive SECIS-element downstream, and C) UGA coding for a selenocysteine in eukaryotes having a SECIS-element in the 3' UTR. Here, the real stop-codon is different from UGA in most cases. Only in rare cases, the real stop-codon is also UGA.



[a] natural amino acid sequence of mouse MsrB
[b] natural mRNA sequence coding for mouse MsrB
[c] natural mRNA can not form the SECIS-element of *E.coli*
[d] new secondary structure of the MsrB mRNA after inserting a SECIS-element of *E.coli*
[e] mutated mRNA sequence having the SECIS-element of *E.coli* at the right position
[f] amino acid sequence encoded by [e]

Figure 3.3: Example of inserting a SECIS-element of *E.coli* (fdhF) in mouse methionine sulfoxide reductase B (MsrB). Due to the changed nucleotide positions in the SECIS-element the encoded amino acids are changed as well. This results in a changed protein, which may lose its function. Sequences are written top down. Changed positions are given in green.

and the efficiency of selenocysteine insertion (i.e. the quality of the SECIS-element). In this chapter, we will focus on an algorithmic solution to this problem of designing a new SECIS-element in the coding region of a protein directly after the UGA. Additionally to the expression of a eukaryotic selenoprotein in *E.coli* described above, this approach can also be applied to the production of a selenocysteine mutant of a protein that naturally contains cysteine. Here, one has to mutate the codon UGU or UGC coding for cysteine to a UGA coding for selenocysteine as well as to design a new SECIS-element following the mutated codon.

3.3 Related Approaches

The corresponding bioinformatic problem is to search for similar proteins under sequence and structure constraints imposed on the mRNA by the SECIS-element. Backofen and colleagues denoted this problem as *mRNA structure optimization (MRSO)* and gave a first solution to it in [BNS02a]. They considered a fixed SECIS-element without allowing any insertions or deletions at the amino acid level and presented a linear time algorithm to solve this problem. In [BNS02b], it was shown that the problem is **NP**-complete, if more complicated RNA secondary structures (i.e. pseudoknots) are considered. For this case, a factor 2 approximation algorithm was given. This approximation just holds, if the similarity functions take only non-negative values, which is not a practical assumption considering real biological data.

Two years later, Bongartz [Bon04] proposed to use the concept of parameterized complexity in order to solve the problem [DF99]. Parameterized complexity offers the possibility to handle many hard problems that are exponential in only a small part of the input, e.g. a parameter k. Then, the problems can be solved in polynomial time when the parameter k is fixed.

In [BFHV05], Blin and colleagues introduced fixed-parameter algorithms for several parameters of MRSO. They started with considering two natural parameters of MRSO, the number of edge crossings and the number of degree three vertices in the implied structure graph (see Definition 3.5.5). Both parameters are believed to be small in most practical applications. Blin *et al.* [BFHV05] showed that MRSO is solvable in polynomial time when either the number of edge crossings or the number of degree three vertices is fixed. Furthermore, they introduced the *page-number* of the secondary structure graph G as the smallest possible partitioning of the edges in G, such that each subset of edges has no edge crossings under the same vertex ordering as G. They showed that MRSO is **NP**-complete even if the implied structure graph has page-number two. Apart from that, Blin *et al.* proved that MRSO is computable in polynomial time, if the implied structure graph has a bounded *cutwidth*. They defined the cutwidth of a graph with n vertices $\{1, ..., n\}$ as its maximum *p-cutwidth* over all $p \in \{1, ..., n-1\}$, where a p-cutwidth is defined as the number of edges that connect vertices in $\{1, ..., p\}$ to vertices in $\{p+1, ..., n\}$.

In [Gur07], Gurski gave a further fixed parameter solution for MRSO on graphs having a bounded *clique-width*, which is the case for a large class of implied structure graphs. As done in [CO00], he defined the clique-width of a graph by the minimal number of labels needed to define it via a composition mechanism for vertex-labeled graphs. During the composition, vertex disjoint union, addition of edges, and relabeling of vertices are valid operations. According to Gurski [Gur07], MRSO is solvable in polynomial time for all structures having an implied structure graph with a bounded clique-width.

3.4 The Computational Problem

In this thesis, we consider an extension of [BNS02a] where we allow insertions and deletions in the amino acid sequence. The reason is that, albeit the SECIS-element is fixed in the mRNA, it is possible to place it on different positions of the amino acid sequence via insertions and deletions of amino acids. This may be necessary if fixed nucleotides in the SECIS-element compete with functional amino acids in the protein (see Figure 3.5).
As we will explain in the following, in this problem insertions and deletions are more complicated than in the usual alignment problems, since an insertion or deletion of an amino acid changes the mapping between the mRNA and the original protein sequence. In addition, we introduce a second optimization criteria: optional base pairs. The reason for this extension is that in the SECIS-element only some part of the structure is fixed. Other elements (like the lower part of the hairpin stem) are not really required, albeit they improve the quality of the SECIS-elements. This is captured by the concept of optional base pairs, which will be introduced in the following as well.

We consider the downstream region of the position where we wish to insert selenocysteine. As input, we have the nucleotide sequence constraints $C = C_1...C_{3n}$ (also called *SECIS constraints*) and the secondary structure T of the SECIS-element as well as the original amino acid sequence $A = A_1...A_n$ of the protein where the selenocysteine is to be inserted. In this numbering, selenocysteine is included at position A_0.

$$\begin{array}{rccccccc}
A & = & A_1 & \dots & A_i & \dots & A_n \\
 & & \sim & & \sim & & \sim \\
A' & = & A'_1 & \dots & A'_i & \dots & A'_n \\
S & = & \overbrace{S_1 S_2 S_3} & \dots & \overbrace{S_{3i-2} S_{3i-1} S_{3i}} & \dots & \overbrace{S_{3n-2} S_{3n-1} S_{3n}} \\
 & & \sim\ \sim\ \sim & & \sim\quad\sim\quad\sim & & \sim\quad\sim\quad\sim \\
C & = & C_1 C_2 C_3 & \dots & C_{3i-2} C_{3i-1} C_{3i} & \dots & C_{3n-2} C_{3n-1} C_{3n}
\end{array}$$

Figure 3.4: Similarity on the nucleotide level as well as on the amino acid level. $S = S_1 ... S_{3n}$ is the mRNA sequence to be searched for. $\sim$ indicates the similarity on both the amino acid ($A \sim A'$) and the nucleotide level ($S \sim C$).

Notation 3.4.1 (SECIS-codon site)
Each codon $C_{3i-2}C_{3i-1}C_{3i}$ of the SECIS constraints with $1 \leq i \leq n$ is also denoted as **SECIS-codon site**.

The process of inserting a SECIS-element poses the problem of finding an appropriate mRNA sequence $S = S_1 ... S_{3n}$. This mRNA sequence must contain a SECIS-element (sequence and structure) at the right position, i.e. is has to fulfill the constraints on the sequence C as well as on its secondary structure T. In addition, we also need to require that the amino acid sequence $A' = A'_1 ... A'_n$ encoded by S has maximum similarity with A. This combination of the similarities on the nucleotide and the amino acid level is shown in Figure 3.4 for the basic case without insertions and deletions. More formally, the problem of designing a SECIS-element can be defined as follows.

Definition 3.4.2 (SECIS-element Design Problem)
Given are an RNA sequence constraints vector $C = C_1 ... C_{3n}$ of length $3n$, an RNA secondary structure T, and an amino acid sequence $A = A_1 ... A_n$ of the protein where the selenocysteine is to be inserted. Then, the **problem of designing a SECIS-element** *is the challenge of finding an RNA sequence $S = S_1 ... S_{3n}$ coding for an amino acid sequence $A' = A'_1 ... A'_n$ with the property that the combination of the similarities of S and C and of A and A' is maximized.*

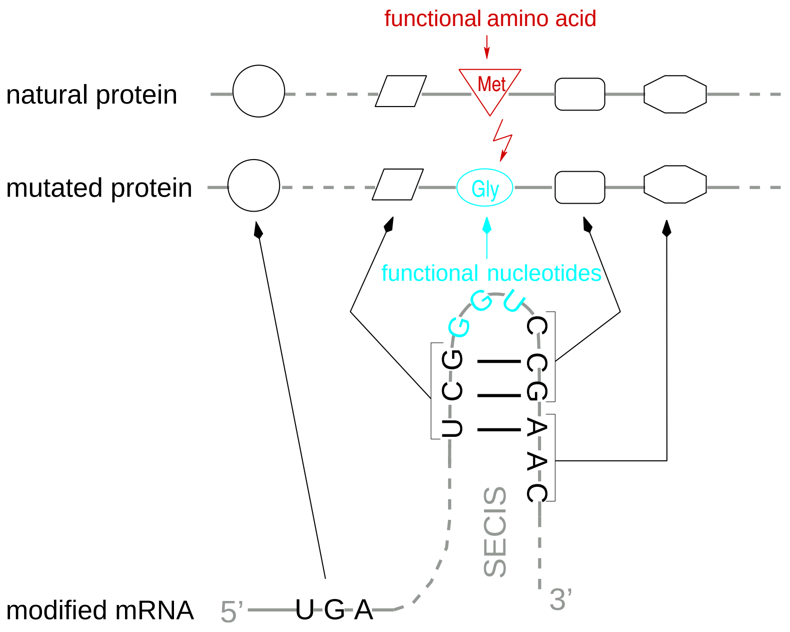


Figure 3.5: Example of contradictory constraints of the nucleotide and the amino acid level. Here, the nucleotides in the hairpin loop are fixed (given in cyan), while the amino acid at the respective position in the protein sequence is fixed to methionine since it is essential for the function of the protein (given in red). A contradiction arises since AUG is the only codon that codes for methionine but it is not consistent with the constraints at the respective nucleotide positions.

Definition 3.4.3 (Similarity Function)
The combined similarity as described in Definition 3.4.2 and shown in Figure 3.4 is formally given by the **similarity function** $\boldsymbol{F_{A_i}^{C_{3i-2}C_{3i-1}C_{3i}}(S_{3i-2}S_{3i-1}S_{3i})}$ *for each position i in the amino acid sequence. Here, the similarity of positions $3i-2$, $3i-1$, and $3i$ of the designed nucleotide sequence depends on both, the similarity to the respective positions in C and to the corresponding position in A.*

The similarity function considers constraints on the nucleotide as well as on the amino acid level. This is needed since some positions of the SECIS sequence C as well as some positions of the amino acid sequence A may be highly conserved and ensure the biological function of the SECIS-element and the protein, respectively. They must not be changed. Therefore, *nucleotide* and *amino acid constraints* penalize or forbid changes at conserved positions. A problem arises if the nucleotide and amino acid constraints are contradictory. An example is given in Figure 3.5.

In this thesis, we show how such contradictions can be solved. We will consider two extensions to the original problem, the insertion and deletion of amino acids and the concept of optional base pairs.

1. Insertions and Deletions. In the original problem, a direct mapping from nucleotide positions to amino acid positions exists, where the codon $S_{3i-2}S_{3i-1}S_{3i}$ on the nucleotide level corresponds to the i-th position on the amino acid level. This changes if one considers *insertions* and *deletions* additionally. Since the SECIS-element is relatively fixed, we consider only insertions and deletions on the amino acid level.
Now, we consider fixed sequence positions $S_{3i-2}S_{3i-1}S_{3i}$ corresponding to the i-th amino acid. Suppose that we have no insertions and deletions so far.

Definition 3.4.4 (Insertion)
An **insertion** *at amino acid position i implies that a new amino acid A'_i is inserted, which does not have a counterpart in A. This implies that*

1. *only the similarity between $S_{3i-2}S_{3i-1}S_{3i}$ and the corresponding part of the SECIS constraints $C_{3i-2}C_{3i-1}C_{3i}$ is measured,*
2. *a penalty for the insertion is added, and*
3. *the mapping from the nucleotide sequence to the original amino acid sequence is changed.*

Therefore for all $j > i$, positions $S_{3j-2}S_{3j-1}S_{3j}$ and A'_j, respectively, have to be compared with the $(j-1)$-th amino acid A_{j-1} of the original protein sequence.

According to definition 3.4.4, at most one amino acid can be inserted per SECIS-codon site. An example is shown in Figure 3.6A. In comparison, a *deletion* is defined as follows and also shown exemplarily in Figure 3.6A.

Definition 3.4.5 (Deletion)
A **deletion** *at amino acid position i implies that codon $S_{3i-2}S_{3i-1}S_{3i}$ has to be compared with the $(i+1)$-th amino acid A_{i+1} and thus A_i is skipped. This means that*

1. *the similarity between $S_{3i-2}S_{3i-1}S_{3i}$ and the corresponding part of the SECIS constraints $C_{3i-2}C_{3i-1}C_{3i}$ as well as the similarity between A_{i+1} and A'_i are measured,*

2. *a penalty for the deletion is added, and*

3. *the mapping from the nucleotide sequence to the original amino acid sequence is changed.*

There could be several deletions per SECIS-codon site. However, since only few insertions and deletions should occur in our designed sequences and since we are restricted to one insertion per SECIS-codon site by the nature of the problem, we confine ourselves to only one deletion per SECIS-codon site. An example that is excluded by this restriction is given in 3.6B.

As a consequence, two consecutive amino acids can be inserted (at two consecutive SECIS-codon sites), but no two consecutive amino acids of the original protein sequence can be deleted. This is due to the fact that we consider only deletions (and insertions) at the amino acid level and not on the nucleotide level. Thus, for each codon $C_{3i-2}C_{3i-1}C_{3i}$ of the SECIS constraints a corresponding codon $S_{3i-2}S_{3i-1}S_{3i}$ in sequence S has to be designed. Therefore, a substitution has to be done at each SECIS-codon site in addition to a possible deletion.

Figure 3.6A shows a correct example of a deletion followed by a substitution at SECIS-codon site 2 whereas, Figure 3.6B exemplifies the incorrect case of two consecutive deletions at SECIS-codon sites 2 and 3.[1]

By the given restriction, it is easier to skip the alignment notation and to represent insertions and deletions directly by a vector.

Definition 3.4.6 (Deletions and Insertions Vector)
The **deletions and insertions vector *t*** *indicates which operation is applied on the SECIS-codon sites. For all i with $1 \leq i \leq n$, it is $t_i \in \{-1, 0, 1\}$. Here, -1 indicates a deletion, $+1$ an insertion, and 0 a simple substition of A_i by another amino acid or itself. The values in t determine the offset that has to be added in order to find the mapping between A'_i and the corresponding position A_j in the original amino acid sequence. Using vector t, index j can be calculated by*

$$j = i - \sum_{k \leq i} t_k.$$

[1]To allow for consecutive deletions (as indicated in Figure 3.6B), one has to allow two or more deletions per SECIS-codon site within the *basic case* (see Equation 3.5).

An example is given in Figure 3.6A. Here, we have $t_1 = 0$, which implies that we have a substitution and compare A'_1 with A_1. For position $i = 2$, we have a deletion indicated by $t_2 = -1$. Hence, we have to compare A'_2 with A_3.

Therefore, we have to consider a modified similarity function $f_i(L_i, d_i, t_i)$. In contrast to $F_{A_i}^{C_{3i-2}C_{3i-1}C_{3i}}(S_{3i-2}S_{3i-1}S_{3i})$, which only depends on the similarity of $S_{3i-2}S_{3i-1}S_{3i}$ and A'_i to $C_{3i-2}C_{3i-1}C_{3i}$ and A_i, respectively, the modified similarity function takes a possible displacement into account additionally. It depends on

- L_i, which represents the codon corresponding to the nucleotides $S_{3i-2}S_{3i-1}S_{3i}$,
- t_i, which is defined above in Definition 3.4.6, and
- d_i, which is the difference between the number of insertions and the number of deletions up to position i, i.e. $d_i = \sum_{j=1}^{i} t_j$, which reflects the relative displacement of the old and the new amino acid sequence to each other.

Definition 3.4.7 (Modified Similarity Function)
The **modified similarity function $f_i(L_i, d_i, t_i)$** *takes three different cases into account. It depends on the value of t_i and is defined as follows.*

$$\begin{aligned}
f_i(L_i, d_i, 0) &= F_{A_{i-d_i}}^{C_{3i-2}C_{3i-1}C_{3i}}(S_{3i-2}S_{3i-1}S_{3i}) \\
f_i(L_i, d_i, +1) &= IP + F^{C_{3i-2}C_{3i-1}C_{3i}}(S_{3i-2}S_{3i-1}S_{3i}) \\
f_i(L_i, d_i, -1) &= \begin{cases} f_i(L_i, d_i, 0) + DP & \text{if there is no constraint for the amino} \\ & \text{acid at position } i - (d_i - t_i) = i - d_{i-1} \\ -\infty & \text{otherwise} \end{cases}
\end{aligned} \tag{3.1}$$

where IP and DP are a penalty for an insertion and a deletion, respectively. In $F^{C_{3i-2}C_{3i-1}C_{3i}}(S_{3i-2}S_{3i-1}S_{3i})$, no amino acid similarity is added since the inserted amino acid A'_i has no counterpart in A to be compared with. Instead, only the similarity of $C_{3i-2}C_{3i-1}C_{3i}$ and $S_{3i-2}S_{3i-1}S_{3i}$ is taken into account.

If a substitution is at position i, i.e. $t_i = 0$, the corresponding nucleotide positions have to be evaluated and the similarity is calculated as hitherto. If an insertion

A)

$$\begin{array}{lccccccc}
A: & A_1 & A_2 & A_3 & A_4 & - & - & A_5 \\
A': & A'_1 & - & A'_2 & A'_3 & A'_4 & A'_5 & A'_6 \\
S: & \overbrace{S_1S_2S_3} & --- & \overbrace{S_4S_5S_6} & \overbrace{S_7S_8S_9} & \overbrace{S_{10}S_{11}S_{12}} & \overbrace{S_{13}S_{14}S_{15}} & \overbrace{S_{16}S_{17}S_{18}} \\
C: & \underbrace{C_1C_2C_3} & \multicolumn{2}{c}{\underbrace{---\quad C_4C_5C_6}} & \underbrace{C_7C_8C_9} & \underbrace{C_{10}C_{11}C_{12}} & \underbrace{C_{13}C_{14}C_{15}} & \underbrace{C_{16}C_{17}C_{18}} \\
t: & 0 & \multicolumn{2}{c}{-1} & 0 & +1 & +1 & 0 \\
\text{SECsite} & 1 & \multicolumn{2}{c}{2} & 3 & 4 & 5 & 6
\end{array}$$

B)

$$\begin{array}{lccccccc}
A: & A_1 & A_2 & A_3 & A_4 & - & - & A_5 \\
A': & A'_1 & - & - & A'_2 & A'_3 & A'_4 & A'_5 \\
S: & \overbrace{S_1S_2S_3} & --- & --- & \overbrace{S_4S_5S_6} & \overbrace{S_7S_8S_9} & \overbrace{S_{10}S_{11}S_{12}} & \overbrace{S_{13}S_{14}S_{15}} \\
C: & \underbrace{C_1C_2C_3} & \underbrace{C_4C_5C_6} & \multicolumn{2}{c}{\underbrace{---\quad C_7C_8C_9}} & \underbrace{C_{10}C_{11}C_{12}} & \underbrace{C_{13}C_{14}C_{15}} & \underbrace{C_{16}C_{17}C_{18}} \\
t: & 0 & -1 & \multicolumn{2}{c}{-1} & +1 & +1 & 0 \\
\text{SECsite} & 1 & 2 & \multicolumn{2}{c}{3} & 4 & 5 & 6
\end{array}$$

Figure 3.6: Allowed and disallowed deletion patterns. A) Possible solution, each deletion must be followed by a substitution. B) Not allowed solution, a deletion (of amino acid A_2) not followed by a substitution. SECsite indicates the number of the SECIS-codon site.

is done at position i, i.e. $t_i = +1$, the similarity is calculated by a combination of a penalty IP for the insertion and the similarity on the nucleotide level. There is no amino acid restriction because inserting each amino acid should be allowed. In contrast, if there is a deletion at the i-th SECIS-codon site, i.e. $t_i = -1$, one has to distinguish between two cases: (i) if there is no constraint for the amino acid at position $i - d_i - 1 = i - d_{i-1}$, which is the corresponding amino acid to be deleted in the original protein sequence, the similarity is calculated the normal way (taking into account the relative displacement d_i of the amino acid sequences, where the deletion at SECIS-codon site i ($t_i = -1$) is included) with an additional penalty DP for the deletion and (ii) if a constraint for the amino acid at position $i - d_i - 1$ exists, deleting this amino acid is not allowed and the similarity function has to be set to $-\infty$.

As mentioned above, only one insertion per SECIS-codon site is possible and therefore, we restrict ourselves to only one deletion per SECIS-codon site as well. Due to this restriction, no two consecutive deletions can be done since the design of a codon $S_{3i-2}S_{3i-1}S_{3i}$ is necessary at every SECIS-codon site. If one wants to allow for two consecutive deletions, these have to be done at one SECIS-codon site and thus, values -2, -3, and so on have to be valid for t_i. However in this thesis, we consider only one deletion per SECIS-codon site.

2. Optional Base Pairs. For the second extension, we now declare some base pairs in the secondary structure as being optional. In contrast to the mandatory pairs, these optional ones are not fixed.

Definition 3.4.8 (Optional Base Pairs)
All base pairs in the secondary structure T that would be of advantage if they form but are not necessary to ensure the function of the SECIS-element are called **optional base pairs**.

An overview of all kinds of base pairs as well as a short description of their meanings is given in Table 3.2. It is described in more detail in the next section.

Due to these two improvements (allowing insertions and deletions and declaring some base pairs as been optional) there are two different and possibly competing values to be maximized: the similarity and the number of realized optional base pairs and not realized unfavorable base pairs. The latter base pairs have to be taken into account since it is of advantage if an unfavorable base pair is not realized, e.g. in case of a functional internal loop.

Name (Label $Lab(v_k, v_l)$)	Meaning	Complementarity condition
base pair (PAIR)	mandatory base pair	$S_k \in S_l^C$
optional base pair (OPT-PAIR)	not necessary pair, but of advantage if formed	$S_k \in \Sigma^B$
G-U pair (GU-PAIR)	mandatory pair that also allows G-U bindings	$S_k \in S_l^{C(G-U)}$
optional G-U pair (OPT-GU-PAIR)	optional pair that also allows G-U bindings	$S_k \in \Sigma^B$
prohibited base pair (PROHI-PAIR)	not allowed pair	$S_k \notin S_l^C$
unfavorable base pair (UNFAV-PAIR)	not necessarily unpaired, but of advantage if not formed	$S_k \in \Sigma^B$

Table 3.2: Different kinds of base pairs (S_k, S_l) and their meanings. Here, the complement set of a variable is denoted by the superscript C. $C(G-U)$ represents the complement set, where G and U are also defined as complementary.

3.5 The Graph Representation of the Problem

In the following, we allow standard Watson-Crick bonds (A-U, C-G) and in some cases when explicitly stated, we consider G and U as complements to each other, too.

Input: According to Definition 3.4.2, we have given

- the target RNA secondary structure T (here, including information about optional base pairs),
- the constraints on the nucleotides $C = C_1...C_{3n}$,
- the constraints on the amino acids $A = A_1...A_n$ (represented by the amino acid sequence of the original protein), and
- the similarity functions $f = \{f_1, ..., f_n\}$ that evaluate both the similarity on the nucleotide as well as on the amino acid level. Each f_i is associated with $\{S_{3i-2}, S_{3i-1}, S_{3i}\}$, $1 \leq i \leq n$, see above.

To formalize the problem, structure T is represented by an edge-labeled graph $G = (V, E, Lab)$:

Definition 3.5.1 (Graph Representation of Structure T)
Secondary structure T is represented by an **edge-labeled graph $G = (V, E, Lab)$** *on 3n vertices with $V = V(G) = \{v_1, ..., v_{3n}\}$. Here, each vertex v_k represents a nucleotide S_k or a position k in the RNA sequence, respectively, and pairs between bases are specified by edges between the respective vertices.*[2] *For every edge $\{v_k, v_l\} \in E = E(G)$, the label $Lab(v_k, v_l)$ indicates the type of the base pair. It is taken from the set {PAIR, OPT-PAIR, GU-PAIR, OPT-GU-PAIR, PROHI-PAIR, UNFAV-PAIR} according to Table 3.2.*

The only label that requires a bit of explanation is $Lab(v_k, v_l)$ = UNFAV-PAIR (unfavorable base pair). This implies that it would be advantageous if there was no base pair. Prohibited base pairs are necessary since the SECIS-element needs specific interior loops. In the following, we assume that $\{v_k, v_l\} \in E(G)$ and $Lab(v_k, v_l)$ = PROHI-PAIR imply that v_k and v_l are not part of a pair. A small example is shown in Figure 3.7A and C.

[2]Here, in contrast to Definition 1.2.1, vertices that represent neighbored nucleotides on the backbone are not connected by an edge.

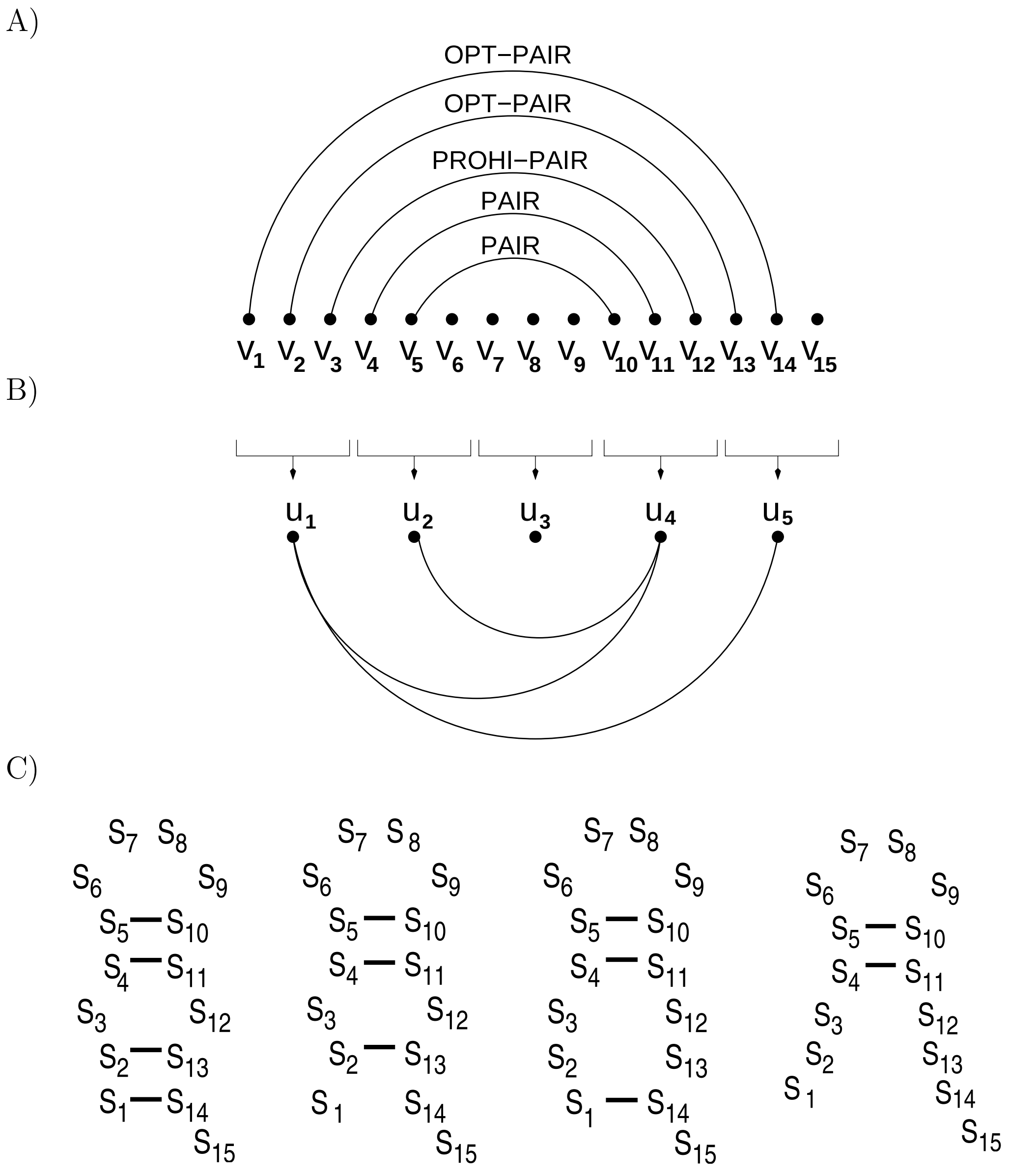

A)
OPT–PAIR
OPT–PAIR
PROHI–PAIR
PAIR
PAIR
V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15
B)
u1 u2 u3 u4 u5
C)
S7 S8 S6 S9 S5 S10 S4 S11 S3 S12 S2 S13 S1 S14 S15

Figure 3.7: Exemplary structure graphs and corresponding structures. A) shows a structure graph with two base pairs, two optional base pairs, and one prohibited base pair (between bases S_3 and S_{12}). B) displays the implied graph of the structure graph given in A). C) illustrates all secondary structures that are allowed concerning to the structure graph in A). Base pairs (S_5, S_{10}) and (S_4, S_{11}) are mandatory and thus exist in each of the structures. The optional base pairs (S_2, S_{13}) and (S_1, S_{14}) can be included but need not, whereas the prohibited base pair between S_3 and S_{12} is not allow and thus must not form in any of the structures.

Notation 3.5.2
The number of all optional pairs, optional G-U pairs, and unfavorable pairs is denoted as ***maxOp****.*

Notation 3.5.3 (Complementarity Condition)
In the following, the complementarity (according to Table 3.2) of all pairs imposed by $E(G)$ is referred to as **complementarity condition***.*

Output: We find vectors

- $S = (S_1, .., S_{3n}) \in \{A, C, G, U\}^{3n}$, such that S_k is assigned to v_k and the complementarity condition of graph G according to $E(G)$ and $Lab(G)$ holds and
- $t = (t_1, ..., t_n) \in \{-1, 0, 1\}^n$ representing the sequence of insertions, deletions, and substitutions

with the feature that they maximize the overall similarity of the assignment

$$\sum_{i=1}^{n} f_i(S_{3i-2}S_{3i-1}S_{3i}, d_i, t_i) = \sum_{i=1}^{n} f_i(L_i, d_i, t_i)$$

(a) over all assignments or

(b) among all assignments having the maximal number of realized optional (G-U) pairs and not realized unfavorable pairs.

The functions $f_i(L_i, d_i, t_i)$ are used as introduced in Definition 3.1, where t_i indicates the operation at SECIS-codon site i (insertion, deletion, or substitution) and d_i represents the sum of all operations done at positions smaller or equal to i, i.e.

$$d_i = \sum_{j=1}^{i} t_j. \tag{3.2}$$

Notation 3.5.4 (SECISDesign)
The above described problems are denoted by (a) $\boldsymbol{SECISDesign_{all}(G, f, C, A)}$ *and (b)* $\boldsymbol{SECISDesign_{opt}(G, f, C, A)}$*.*

Additional Notations: G is referred to as the structure graph on the mRNA level. Accordingly, we need such a graph on the amino acid level as well.

Definition 3.5.5 (Implied Graph)
Given an mRNA structure graph G, we define the **implied graph $\boldsymbol{G^{impl}}$** *as a graph on the vertices $V(G^{impl}) = \{u_1, ..., u_n\}$ with*

$$E(G^{impl}) = \left\{ \{u_i, u_j\} \;\middle|\; \begin{array}{r} \exists r \in \{3i-2, 3i-1, 3i\} \wedge \exists s \in \{3j-2, 3j-1, 3j\} : \\ \{v_r, v_s\} \in E(G) \end{array} \right\}.$$

Note that in the implied graph every node has at most degree 3 if the nodes of the input graph G have at most degree 1. (This holds in our problem since we have at most one (G-U) base pair, one optional (G-U) base pair, one prohibited base pair, or one unfavorable base pair per node.) Hence, up to three edges can emerge from a node in the implied graph. An example is given in Figure 3.7A and B.

Notation 3.5.6
Independent of the subscript or superscript, $\boldsymbol{u}$ denotes a vertex from $V(G^{impl})$ and $\boldsymbol{v}$ denotes a vertex from $V(G)$.

Definition 3.5.7 (Valid Codons, Valid Codon Sequence)
Two **codons** *L_i and L_j are said to be* **valid for $\boldsymbol{\{u_i, u_j\}}$ w.r.t. $\boldsymbol{E(G)}$** *if the corresponding nucleotides $S_{3i-2}S_{3i-1}S_{3i}$ and $S_{3j-2}S_{3j-1}S_{3j}$ satisfy the complementarity condition imposed by $E(G)$.*
Analogously, a **codon sequence** *$L_1...L_n$ is said to be* **valid for $\boldsymbol{E(G)}$** *iff the corresponding nucleotides $\{S_{3i-2}S_{3i-1}S_{3i}\}_{i=1}^{n}$ satisfy the complementarity condition imposed by $E(G)$.*

3.6 SECISDesign - The Algorithm

We present a polynomial time recursive algorithm that solves SECISDesign$_{all}$ and SECISDesign$_{opt}$ when G^{impl} is outer-planar (see Definition 3.6.1) *and* every node in G has at most degree one. The hairpin shape of the SECIS-element is captured by the outer-planarity of G^{impl}. Our algorithm is based on a recurrence relation that we introduce in this section.

Definition 3.6.1 (Outer-Planar Graph)
An undirected graph is said to be **outer-planar** *if, when drawn, it satisfies the following conditions:*

- *all the vertices lie on a line,*
- *all the edges lie on one side of the line, and*
- *two edges that intersect in the drawing do so only at their end points.*

We fix $u_1, ..., u_n$ as the ordering of the vertices from left to right on a line in an outer-planar embedding of G^{impl}. Let f_i be the function associated with u_i. Having fixed an embedding of the graph on a line, we do not distinguish between a vertex and its index. That is:

Notation 3.6.2
The interval $\boldsymbol{[i...i+k]}$ *denotes the set of vertices* $\{u_i, ..., u_{i+k}\}$.

Notation 3.6.3
Given an interval $[i...i+k]$,

- $\boldsymbol{E(G^{impl})|_{[i...i+k]}}$ *is the set of edges of the induced subgraph of* G^{impl} *on* $\{u_i, ..., u_{i+k}\}$, *and*
- $\boldsymbol{E(G)|_{[i...i+k]}}$ *denotes the edge set of the induced subgraph of graph* G *on* $\{v_{3j-2}, v_{3j-1}, v_{3j} | i \leq j \leq i+k\}$.

Now, we define the notation of compatibility between codons L_i and L_j assigned to vertices u_i and u_j, respectively. Some additional notations are given in Table 3.3.

Notation 3.6.4 (Compatibility)
The compatibility with respect to the complementarity condition in $E(G)$ *is denoted by* $\equiv_{\boldsymbol{E(G)}}$.

$ro(L_i...L_{i+k})$	number of ***r***ealized ***o***ptional pairs, realized optional G-U pairs, and unrealized unfavorable pairs in $E(G)\vert_{[i...i+k]}$
s^I	number of insertions at and between positions $i+1$ and $i+k$
s^D	number of deletions at and between positions $i+1$ and $i+k$
s	$s = s^I - s^D = d_{i+k} - d_i$, i.e. the difference of insertions and deletions at and between positions $i+1$ and $i+k$ ('in***s***ide' the interval)
l	$l = d_i = \sum_{x=1}^{i} t_i$, i.e. the difference of insertions and deletions up to position i (at all positions $\leq i$, i.e. '***l***eft' of i)
$\mathcal{V}(a, b, len)$	set of vectors with length len having a coordinates with value $+1$, b with value -1, and $len - (a+b)$ coordinates being 0. It is necessary that $a + b \leq len$.
$t(x : y)$	part of vector t that starts at the x-th coordinate and ends at the y-th coordinate and, thus, has length $y - x + 1$.

Table 3.3: Additional notations used during SECISDesign. They correspond to an interval $[i...i+k]$ if nothing further is mentioned.

Definition 3.6.5 (Compatible Codons)
For $1 \leq i, j \leq n$, we define

$$\boldsymbol{(i, L_i) \equiv_{E(G)} (j, L_j)} = \begin{cases} \textit{true} & \textit{if } \{u_i, u_j\} \notin E(G^{impl}) \\ \textit{true} & \textit{if } \{u_i, u_j\} \in E(G^{impl}) \textit{ and} \\ & L_i \textit{ and } L_j \textit{ are valid for } \{u_i, u_j\} \textit{ w.r.t. } E(G) \\ \textit{false} & \textit{otherwise} \end{cases}$$

As mentioned above, we allow only one insertion or deletion per position. Thus for a given interval $[i...i+k]$, it is necessary that

$$|l| \leq i \quad \wedge \quad |s| \leq k. \tag{3.3}$$

Definition 3.6.6 ($SPLIT_k^s$)
SPLIT$_k^s$ *is defined as the set of tupels (s^I, s^D) that fulfill*

$$s^I - s^D = s, \qquad s^I + s^D \leq k, \qquad 0 \leq s^I, \quad \textit{and} \quad 0 \leq s^D.$$

Now, we define the central function of SECISDesign $w_{i+k}^i(L_i, L_{i+k}, m, l, s)$, which can be implemented by dynamic programming (see Theorem 3.6.9).

Definition 3.6.7 (Central Function of SECISDesign)
$\boldsymbol{w_{i+k}^i(L_i, L_{i+k}, m, l, s)}$ *gives the maximal similarity of the designed sequence to the nucleotide and amino acid constraints between positions i and $i+k$, where L_i is the codon at position i and L_{i+k} is the codon at position $i+k$ of the new sequence. The number of realized optional and not realized unfavorable pairs $ro(L_i...L_{i+k})$ in the interval $[i...i+k]$ is fixed to m. l, s, s^I, and s^D are defined as given in Table 3.3. Thus,*

$$w_{i+k}^i(L_i, L_{i+k}, m, l, s)$$
$$= \max_{\substack{L_{i+1}...L_{i+k-1} \\ t(i+1:i+k) \in \mathcal{V}(s^I, s^D, k) \\ (s^I, s^D) \in SPLIT_k^s}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) \;\middle|\; \begin{array}{l} L_i...L_{i+k} \textit{ valid for} \\ E(G)|_{[i...i+k]} \textit{ and} \\ ro(L_i...L_{i+k}) = m \end{array} \right\} \tag{3.4}$$

where $l + \sum_{x=i+1}^{j} t_x = \sum_{x=1}^{j} t_x = d_j$ as given in Equation 3.2. If the set over which the maximum is taken is empty, the value of the function is considered to be $-\infty$.

Here, l is used to find the relative position of the original amino acid sequence to the new one resulting from possible insertions and deletions before and at i, which is important to the similarity function.[3] Parameter s indicates the difference of insertions and deletions at and between positions $i+1$ and $i+k$. It contributes to the correct splitting of the interval, which is explained in the following. Additionally to l, $\sum_{x=i+1}^{j} t_x$ is needed to find the relative position of the original amino acid sequence A and the new amino acid sequence A' (encoded by the designed mRNA sequence S) to each other in all considered intervals.

[3] Notice: In contrast to position $i+k$, the similarity at position i is not included in w_{i+k}^i.

As an example, we consider the structure graph G and its corresponding implied structure graph G^{impl} shown in Figure 3.7. According to Definition 3.6.7, $w_5^1(L_1, L_5, m, l, s)$ results from

$$\max_{\substack{L_2, L_3, L_4 \\ t(2:5) \in \mathcal{V}(s^I, s^D, 4) \\ (s^I, s^D) \in SPLIT_4^s}} \left\{ \begin{array}{l|l} f_2(L_2, l + t_2, t_2)+ & L_1...L_5 \text{ valid for} \\ f_3(L_3, l + t_2 + t_3, t_3)+ & E(G)|_{[1...5]} \text{ and} \\ f_4(L_4, l + t_2 + t_3 + t_4, t_4)+ & ro(L_1...L_5) = m \\ f_5(L_5, l + t_2 + t_3 + t_4 + t_5, t_5) & \end{array} \right\}.$$

If we set $L_1 = ACG$, $L_5 = GUC$, $m = 2$, and $t = (+1, 0, 0, +1, 0)$, it follows that, in case of $w_5^1(L_1, L_5, m, l, s)$, parameters $l = 1$ and $s = 1$. Thus for these given values, it is

$$w_5^1(ACG, GUC, 2, 1, 1) = \max_{L_2, L_3, L_4} \left\{ \begin{array}{l|l} f_2(L_2, 1, 0)+ & L_1...L_5 \text{ valid for} \\ f_3(L_3, 1, 0)+ & E(G)|_{[1...5]} \text{ and} \\ f_4(L_4, 2, 1)+ & ro(L_1...L_5) = 2 \\ f_5(L_5, 2, 0) & \end{array} \right\}.$$

The functions w_{i+k}^i form the central part of our algorithm because at first they treat both objective functions separately (maximizing the similarity and maximizing the number of realized optional and not realized unfavorable pairs) and one can combine them afterwards, as follows, to get the value of interest:

1. SECISDesign$_{all}$: Maximize the similarity overall. Among all assignments having the maximal similarity, choose the one with the highest number of realized optional and not realized unfavorable base pairs. The value of interest is

$$\max_{L_1, L_n, m} \left\{ \max_{|l| \leq 1, |s| \leq n-1} \{ w_n^1(L_1, L_n, m, l, s) + f_1(L_1, l, l) \} \right\}.$$

2. SECISDesign$_{opt}$: Maximize the number of realized optional base pairs and not realized unfavorable pairs first. Among all assignments that realize this maximal number choose the assignment having the highest similarity. Therefore, the value of interest is

$$\max_{L_1,L_n}\left\{\max_{|l|\leq 1,|s|\leq n-1}\{w_n^1(L_1,L_n,m_{max},l,s)\;+\;f_1(L_1,l,l)\}\right\},$$

where $m_{max} \leq maxOp$ is the maximal value that can be adopted by m without violating any conditions.

Initialization. The equation for the basic case of w, where only two neighbored positions i and $i+1$ are considered, is the following:

$\forall m : 0 \leq m \leq maxOp \quad \forall l : |l| \leq i \quad \forall s : |s| \leq 1$

$$w^i_{i+1}(L_i, L_{i+1}, m, l, s) = \begin{cases} f_{i+1}(L_{i+1}, l+s, s) & \text{if } (i, L_i) \equiv_{E(G)} (i+1, L_{i+1}) \\ & \quad \wedge\, ro(L_i L_{i+1}) = m \\ -\infty & \text{otherwise} \end{cases} \tag{3.5}$$

It shows that we allow at most one insertion or deletion per position. As mentioned before, it is not possible to prevent two or more insertions at two successive positions straightforwardly. Thus, if two consecutive insertions are undesirable, the insertion-penalty has to be chosen appropriately.

Recursion. In the recursion, we solve the problem on an interval $[i...i+k]$ by splitting it into two parts and solving the resulting subproblems.

Definition 3.6.8 (Maximal Edge)
If there is no edge between i and any vertex in $[i...i+k-1]$, the interval $[i...i+k]$ is split into $[i...i+1]$ and $[i+1...i+k]$. Otherwise, we choose the farthest vertex p in $[i...i+k-1]$ which is adjacent to i. The edge $\{i,p\}$ is called the **maximal edge** *in $E(G^{impl})|_{[i...i+k-1]}$. Then, the interval is split into $[i...p]$ and $[p...i+k]$ (see Figure 3.8). Hence, we define* ***next(i, i + k)*** *for $k \geq 2$ by*

$$next(i, i+k) = \begin{cases} i+r & \text{if } \{i, i+r\} \text{ is maximal in } E(G^{impl})|_{[i...i+k-1]} \\ i+1 & \text{otherwise.} \end{cases}$$

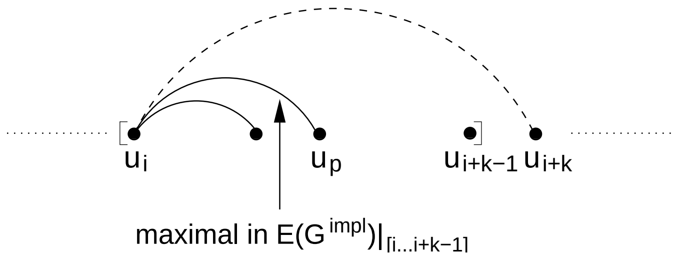


Figure 3.8: Illustration of $next(i, i+k)$.

Theorem 3.6.9 (Recurrence)
Let (G, f, C, A) be an instance of SECISDesign$_{all}$ or SECISDesign$_{opt}$. For $k \geq 2$, $1 \leq i \leq n-k$, $|l| \leq i$, $s = s^I - s^D$ with $|s| \leq k$, $0 \leq m \leq maxOp$, let $p = next(i, i+k)$. Then

$$w^i_{i+k}(L_i, L_{i+k}, m, l, s) =$$

$$\begin{cases} -\infty & \text{if } (i, L_i) \not\equiv_{E(G)} (i+k, L_{i+k}) \\ \max\limits_{\substack{L_p \\ m' + m'' + opt^{L_i}_{L_{i+k}} = m \\ m', m'' \geq 0}} \quad \max\limits_{\substack{s' + s'' = s \\ |s'| \leq p - i \\ |s''| \leq i + k - p}} \begin{pmatrix} w^i_p(L_i, L_p, m', l, s') + \\ w^p_{i+k}(L_p, L_{i+k}, m'', l + s', s'') \end{pmatrix} & \\ & \text{if } (i, L_i) \equiv_{E(G)} (i+k, L_{i+k}) \end{cases}$$

where $opt^{L_i}_{L_{i+k}}$ is the number of realized optional (G-U) base pairs and not realized unfavorable base pairs between codon L_i and codon L_{i+k}.[4] m' and m'' are the numbers of realized optional pairs and not realized unfavorable pairs between all codons L_j with $i \leq j \leq p$ (for m') and with $p \leq j \leq i+k$ (for m''), respectively. s' and s'' specify the differences of insertions and deletions at and between positions $i+1$ and p and between positions $p+1$ and $i+k$, respectively.

Before proofing this theorem, we give a small example to make its recurrence equation clear. Given the structure graph G and its corresponding implied graph G^{impl} already shown in Figure 3.7, we aim at finding a value for $w^1_5(L_1, L_5, m, l, s)$. To find the best value of function w^1_5, we have to maximize over all possible assignments of L_1, L_5, m, l, and s.

[4] Note that in contrast, $ro(L_i ... L_{i+k})$ denotes the number of realized optional (G-U) pairs and not realized unfavorable pairs between all codons L_j with $i \leq j \leq i+k$.

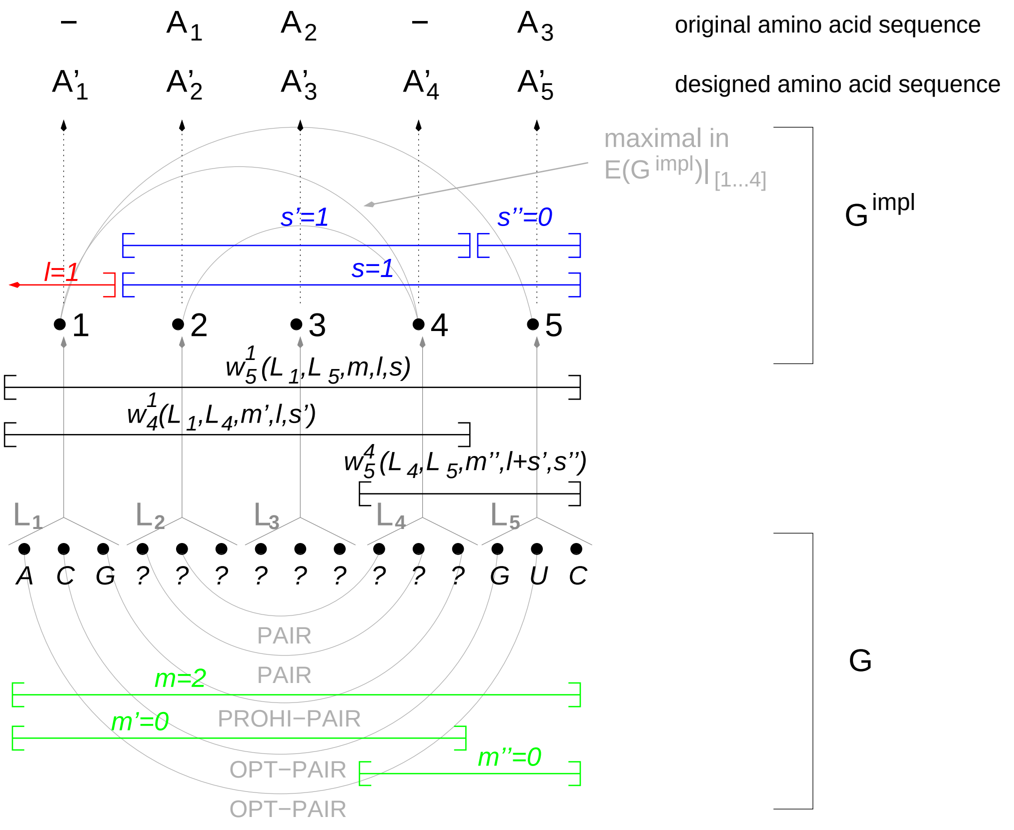


Figure 3.9: Graphical interpretation of the recurrence of Theorem 3.6.9. On the top of the picture, the two amino acid sequences and their relation to each other are shown. The implied graph G^{impl} and the corresponding structure graph G (already introduced in Figure 3.7) are given below. Their edges are drawn in gray in the background. We set $L_1 = ACG$, $L_5 = GUC$, $m = 2$, and $s = 1$ exemplarily. The complete interval [1...5] is split up into two intervals. As a consequence, parameters m and s are also split up. Their corresponding intervals and values are indicated in green and blue, respectively. Apart from that, we set $l = 1$ exemplarily. Its corresponding interval is shown in red. Thus, function $w_5^1(L_1, L_5, m, l, s)$ can be evaluated by adding $w_4^1(L_1, L_4, m', l, s')$ and $w_5^4(L_4, L_5, m'', l + s', s'')$. These two functions overlapp at position 4, but only in function w_5^4 the similarity of L_4 is taken into account (see Equation 3.5).

Here, we are only interested in one precise value, $w_5^1(ACG, GUC, 2, 1, 1)$, which is schematically demonstrated in Figure 3.9. Since graph G includes only two optional base pairs, m has to be lower or equal to 2. We set $m = 2$. Furthermore, we set $l = 1$ exemplarily. This implicates that the difference of insertions and deletions before and at the first codon is 1. Since there is no prior codon, $l = 1$ implicates that an insertion is done at position 1, i.e. the amino acid A'_1 corresponding to the first codon is inserted compared to the original amino acid sequence. Apart from that, we set $s = 1$, which indicates that the difference of insertions and deletions at and between positions 2 and 5 is 1.
The interval [1...5] can be split at the edge $\{1, 4\}$, which is the maximal edge in $E(G^{impl})|_{[1...4]}$. Thus, it is $next(1, 5) = 4$. Since it is $(1, ACG) \equiv_{E(G)} (5, GUC)$, it follows that

$$w_5^1(ACG, GUC, 2, 1, 1) =$$

$$\max_{\substack{L_4 \\ m' + m'' + opt_{L_5}^{L_1} = 2 \\ m', m'' \geq 0}} \quad \max_{\substack{s' + s'' = 1 \\ |s'| \leq 3 \\ |s''| \leq 1}} \left(\begin{array}{r} w_4^1(ACG, L_4, m', 1, s') + \\ w_5^4(L_4, GUC, m'', 1 + s', s'') \end{array} \right)$$

Since the interval [1...5] is split into two intervals, the number of realized optional (G-U) pairs and unfavorable pairs m has also to be split up into m', m'', and $opt_{L_5}^{L_1} = opt_{GUC}^{ACG} = 2$. We set m' and m'' to 0 since there are no optional pairs or unfavorable pairs between codons L_1, L_2, L_3, and L_4 and between codons L_4 and L_5, respectively. Thus, it is

$$w_5^1(ACG, GUC, 2, 1, 1) = \max_{\substack{L_4 \\ s' + s'' = 1 \\ |s'| \leq 3 \\ |s''| \leq 1}} \left(\begin{array}{r} w_4^1(ACG, L_4, 0, 1, s') + \\ w_5^4(L_4, GUC, 0, 1 + s', s'') \end{array} \right)$$

s has to be split into s' and s'', i.e. the difference of insertions and deletions at and between positions 2 and 4 and the difference of insertions and deletions at position 5, respectively. In the example shown in Figure 3.9, we set $s' = 1$ and $s'' = 0$. For this case, we have

$$w_5^1(ACG, GUC, 2, 1, 1) = \max_{L_4} \left(w_4^1(ACG, L_4, 0, 1, 1) + w_5^4(L_4, GUC, 0, 2, 0) \right).$$

Proof of Theorem 3.6.9

In the first case when $(i, L_i) \not\equiv_{E(G)} (i+k, L_{i+k})$, the result is true since the maximum over an empty set is $-\infty$. Thus, we just have to consider the second case where $(i, L_i) \equiv_{E(G)} (i+k, L_{i+k})$. Let $k \geq 2$ and the other conditions of Theorem 3.6.9 be fulfilled. We wish to evaluate the case of $(i, L_i) \equiv_{E(G)} (i+k, L_{i+k})$, i.e.

$$w^i_{i+k}(L_i, L_{i+k}, m, l, s) =$$

$$\max_{\substack{L_p \\ m'+m''+opt^{L_i}_{L_{i+k}}=m \\ m', m'' \geq 0}} \quad \max_{\substack{s'+s''=s \\ |s'| \leq p-i \\ |s''| \leq i+k-p}} \left(\begin{array}{l} w^i_p(L_i, L_p, m', l, s') \\ \qquad +w^p_{i+k}(L_p, L_{i+k}, m'', l+s', s'') \end{array} \right) \tag{3.6}$$

where $w^i_p(L_i, L_p, m', l, s')$ and $w^p_{i+k}(L_p, L_{i+k}, m'', l+s', s'')$ are defined as given in Equation 3.4. We have to show that Equation 3.6 equals

$$\max_{\substack{L_{i+1}...L_{i+k-1} \\ t(i+1:i+k) \in \mathcal{V}(s^I, s^D, k) \\ (s^I, s^D) \in SPLIT^s_k}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) \;\middle|\; \begin{array}{l} L_i...L_{i+k} \text{ valid for} \\ E(G)|_{[i...i+k]} \text{ and} \\ ro(L_i...L_{i+k}) = m \end{array} \right\} \tag{3.7}$$

Equally to $s = s^I - s^D$, s' and s'' can be split into $s'^I - s'^D$ and $s''^I - s''^D$, respectively. Thus, due to Equation 3.4, it is

$$w^i_p(L_i, L_p, m', l, s') =$$

$$\max_{\substack{L_{i+1}...L_{p-1} \\ t(i+1:p) \in \mathcal{V}(s'^I, s'^D, p-i) \\ (s'^I, s'^D) \in SPLIT^{s'}_{p-i}}} \left\{ \sum_{i<j\leq p} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) \;\middle|\; \begin{array}{l} L_i...L_p \text{ valid for} \\ E(G)|_{[i...p]} \text{ and} \\ ro(L_i...L_p) = m' \end{array} \right\} \tag{3.8}$$

and

$$w^p_{i+k}(L_p, L_{i+k}, m'', l+s', s'') =$$

$$\max_{\substack{L_{p+1}...L_{i+k-1} \\ t(p+1:i+k)\in\mathcal{V}(s''^I, s''^D, i+k-p) \\ (s''^I, s''^D)\in SPLIT^{s''}_{i+k-p}}} \left\{ \sum_{p<j\leq i+k} f_j(L_j, l+s'+\sum_{x=p+1}^{j} t_x, t_j) \,\middle|\, \begin{array}{l} L_p...L_{i+k} \text{ valid for } E(G)|_{[p...i+k]} \text{ and} \\ ro(L_p...L_{i+k}) = m'' \end{array} \right\} \tag{3.9}$$

To combine Equations 3.8 and 3.9, we have to show that all possible vector parts $t(i+1:i+k) \in \mathcal{V} = \mathcal{V}(s^I, s^D, k)$ for all possible s^I and s^D are included in the combination of all possible vector parts $t(i+1:p) \in \mathcal{V}' = \mathcal{V}(s'^I, s'^D, p-i)$ and $t(p+1:i+k) \in \mathcal{V}'' = \mathcal{V}(s''^I, s''^D, i+k-p)$ on all possible assignments of s'^I, s''^I, s'^D and s''^D and vice versa, i.e. that

$$\bigcup_{\substack{(s^I, s^D) \\ \in SPLIT^s_k}} \mathcal{V} = \bigcup_{\substack{s'+s''=s \\ |s'|\leq p-i \\ |s''|\leq i+k-p}} \left(\bigcup_{\substack{(s'^I, s'^D) \\ \in SPLIT^{s'}_{p-i}}} \mathcal{V}' \times \bigcup_{\substack{(s''^I, s''^D) \\ \in SPLIT^{s''}_{i+k-p}}} \mathcal{V}'' \right) \tag{3.10}$$

If we combine the vector parts $t(i+1:p) \in \mathcal{V}' = \mathcal{V}(s'^I, s'^D, p-i)$, which have length $p-i$, and the vector parts $t(p+1:i+k) \in \mathcal{V}'' = \mathcal{V}(s''^I, s''^D, i+k-p)$, which have length $i+k-p$, to the vector parts $t(i+1:i+k) = t(i+1:p)t(p+1:i+k) \in \mathcal{V}$, which have length k, it is $s^I = s'^I + s''^I \geq 0$ and $s^D = s'^D + s''^D \geq 0$. Furthermore, if $(s'^I, s'^D) \in SPLIT^{s'}_{p-i}$ and $(s''^I, s''^D) \in SPLIT^{s''}_{i+k-p}$, it holds $(s^I, s^D) \in SPLIT^{s'+s''}_k$ since

$$\begin{aligned} s^I - s^D &= (s'^I + s''^I) - (s'^D + s''^D) \\ &= (s'^I - s'^D) + (s''^I - s''^D) \\ &= s' + s'' \end{aligned}$$

and

$$\begin{aligned} s^I + s^D &= (s'^I + s''^I) + (s'^D + s''^D) \\ &= (s'^I + s'^D) + (s''^I + s''^D) \\ &\leq (p-i) + (i+k-p) \\ &= k. \end{aligned}$$

If $(s'^I, s'^D) \in SPLIT^{s'}_{p-i}$, it follows from Definition 3.6.6 that $|s'| \leq p - i$ since

$$\begin{aligned}
SPLIT^{s'}_{p-i} \; &\Leftrightarrow \; (s'^I - s'^D = s') \wedge (s'^I + s'^D \leq p - i) \wedge (0 \leq s'^I, s'^D) \\
&\Leftrightarrow \quad |s'| \; = |s'^I - s'^D| \\
&\qquad\qquad = |s'^I + (-s'^D)| \\
&\qquad\qquad \leq |s'^I| + |-s'^D| \\
&\qquad\qquad = s'^I + s'^D \text{ since } 0 \leq s'^I, s'^D \\
&\qquad\qquad \leq p - i.
\end{aligned}$$

Analogously, it can be shown that $|s''| \leq i + k - p$ if $(s''^I, s''^D) \in SPLIT^{s''}_{i+k-p}$.

Thus, we have proven Equation 3.10 since

$$\begin{aligned}
&\bigcup_{\substack{s' + s'' = s \\ |s'| \leq p - i \\ |s''| \leq i + k - p}} \left(\bigcup_{\substack{(s'^I, s'^D) \\ \in SPLIT^{s'}_{p-i}}} \mathcal{V}' \times \bigcup_{\substack{(s''^I, s''^D) \\ \in SPLIT^{s''}_{i+k-p}}} \mathcal{V}'' \right) \\
&\qquad = \bigcup_{s'+s''=s} \left(\bigcup_{\substack{(s'^I, s'^D) \\ \in SPLIT^{s'}_{p-i}}} \mathcal{V}' \times \bigcup_{\substack{(s''^I, s''^D) \\ \in SPLIT^{s''}_{i+k-p}}} \mathcal{V}'' \right) \\
&\qquad = \bigcup_{s'+s''=s} \left(\bigcup_{\substack{(s^I, s^D) \\ \in SPLIT^{s'+s''}_{k}}} \mathcal{V} \right) \\
&\qquad = \bigcup_{\substack{(s^I, s^D) \\ \in SPLIT^{s}_{k}}} \mathcal{V}.
\end{aligned}$$

We have shown that each possible vector part $t(i + 1 : i + k)$ of insertions and deletions is taken into account. A further point to be mentioned is the correct calculation of the similarity. The numbers of insertions and deletions in the interval $[i + 1...p]$ result in a further relative displacement (apart from l) of the original to the new amino acid sequence. This displacement is the only effect the intervals

$[i+1...p]$ and $[p+1...i+k]$ have on each other, which has to be considered during the calculation of the similarity within $[p+1...i+k]$. This is achieved through the similarity function $f_x = f_x(L_x, \sum_{j\leq x} t_j, t_x)$ taking into account all insertions and deletions made before and at x. In our case, this is done by s', which equals

$$s' = \sum_{x=i+1}^{p} t_x.$$

Taking this into account, the similarity can be calculated separately in both intervals since the similarity function is additive.

Further considering Equations 3.8, 3.9, and 3.10 and the combination of vector parts $t(i+1:p) \in \mathcal{V}(s'^I, s'^D, p-i)$ and vector parts $t(p+1:i+k) \in \mathcal{V}(s''^I, s''^D, i+k-p)$ described above, Equation 3.6 can be reformulated to

$$\max_{\substack{L_{i+1}...L_p...L_{i+k-1} \\ m'+m''+opt_{L_{i+k}}^{L_i} = m \\ m',m'' \geq 0 \\ s'+s''=s \\ t(i+1:p)\in\mathcal{V}(s'^I,s'^D,p-i) \\ (s'^I,s'^D)\in SPLIT_{p-i}^{s'} \\ t(p+1:i+k)\in\mathcal{V}(s''^I,s''^D,i+k-p) \\ (s''^I,s''^D)\in SPLIT_{i+k-p}^{s''}}} \left\{ \begin{array}{l} \displaystyle\sum_{i<j\leq p} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) + \\ \displaystyle\sum_{p<j\leq i+k} f_j(L_j, l + s' + \sum_{x=p+1}^{j} t_x, t_j) \end{array} \right| \left. \begin{array}{l} L_i...L_p \text{ valid for } E(G)|_{[i...p]} \\ \text{and } ro(L_i...L_p) = m' \\ L_p...L_{i+k} \text{ valid for } E(G)|_{[p...i+k]} \\ \text{and } ro(L_p...L_{i+k}) = m'' \end{array} \right\} \qquad (3.11)$$

$$= \max_{\substack{L_{i+1}...L_{i+k-1} \\ m'+m''+opt_{L_{i+k}}^{L_i} = m \\ m',m'' \geq 0 \\ t(i+1:i+k)\in\mathcal{V}(s^I,s^D,k) \\ (s^I,s^D)\in SPLIT_k^{s}}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) \right| \left. \begin{array}{l} L_i...L_p \text{ valid for } E(G)|_{[i...p]} \\ \text{and } ro(L_i...L_p) = m' \\ L_p...L_{i+k} \text{ valid for } E(G)|_{[p...i+k]} \\ \text{and } ro(L_p...L_{i+k}) = m'' \end{array} \right\}$$

Since we consider the case where $(i, L_i) \equiv_{E(G)} (i+k, L_{i+k})$, we can reason that $L_i...L_p...L_{i+k}$ is valid for $E(G)|_{[i...i+k]}$ if $L_i...L_p$ is valid for $E(G)|_{[i...p]}$ and $L_p...L_{i+k}$ is valid for $E(G)|_{[p...i+k]}$. Furthermore, the number of realized optional pairs, optional G-U pairs, and not realized unfavorable pairs between the two codons L_i and L_{i+k} is $opt^{L_i}_{L_{i+k}}$ (fixed for fixed L_i and L_{i+k}). The compatibility within the regions $[i...p]$ and $[p...i+k]$ holds by assumption. The compatibility of vertex pair (u_r, u_s) with $r \in [i...p-1]$ and $s \in [p+1...i+k]$ holds since $\{i, p\}$ is a maximal edge and there are no edges in $E(G^{impl})$ between r and s (and no corresponding edges in $E(G)$). Therefore, $ro(L_i...L_{i+k})$ can be identified by adding $ro(L_i...L_p)$, $ro(L_p...L_{i+k})$, and $opt^{L_i}_{L_{i+k}}$. Thus, Equation 3.11 can be further reformulated to

$$= \max_{\substack{L_{i+1}...L_{i+k-1} \\ t(i+1:i+k) \in \mathcal{V}(s^I, s^D, k) \\ (s^I, s^D) \in SPLIT^s_k \\ m' + m'' + opt^{L_i}_{L_{i+k}} = m \\ m', m'' \geq 0}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) \;\middle|\; \begin{array}{l} L_i...L_{i+k} \text{ valid for} \\ E(G)|_{[i...i+k]} \text{ and} \\ ro(L_i...L_{i+k}) \\ = m' + m'' + opt^{L_i}_{L_{i+k}} \end{array} \right\}$$

$$= \max_{\substack{L_{i+1}...L_{i+k-1} \\ t(i+1:i+k) \in \mathcal{V}(s^I, s^D, k) \\ (s^I, s^D) \in SPLIT^s_k}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l + \sum_{x=i+1}^{j} t_x, t_j) \;\middle|\; \begin{array}{l} L_i...L_{i+k} \text{ valid for} \\ E(G)|_{[i...i+k]} \text{ and} \\ ro(L_i...L_{i+k}) = m \end{array} \right\}$$

Thus, the two quantities of Equations 3.6 and 3.7 are equal because the maximum for the problem in the region $[i...i+k]$ for fixed L_i, L_p, L_{i+k}, m', l and s' can be obtained by finding $w^i_p(L_i, L_p, m', l, s')$ (maximum in the region $[i...p]$) and $w^p_{i+k}(L_p, L_{i+k}, m - m' - opt^{L_i}_{L_{i+k}}, l + s', s - s')$ (maximum in the region $[p...i+k]$). To find the maximum in $[i...i+k]$ with only L_i and L_{i+k} fixed, we have to maximize over all possible assignments of L_p, m' and s', which is done due to Equation 3.6.

□

REMARK 3.6.10 *The similarity at the left boundary of the interval $[i...i+k]$ is not included in $w^i_{i+k}(L_i, L_{i+k}, m, l, s)$. If it is included, it will be added twice when combining neighbored intervals. The left boundary only suffices to keep the codon on this position to combine the subproblems properly.*

Complexity of the Algorithm. We present the properties of the algorithm at level $[i...i+k]$ of the recursion. Recall that our goal is to compute $w^i_{i+k}(L_i, L_{i+k}, m, l, s)$ for all choices of L_i, L_{i+k}, m, l, s, corresponding assignments of codons and insertions and deletions. Therefore, $O(n^2)$ (to be exact: $64^2 * maxOp * (2i+1) * (2k+1)$) different cases have to be considered. Let $p = next(i, i+k)$. Since each vertex in G^{impl} has, at most, degree 3, p can be found in not more than 3 steps. If $w^i_p(L_i, L_p, m', l, s')$ and $w^p_{i+k}(L_p, L_{i+k}, m - opt^{L_i}_{L_{i+k}} - m', l+s', s-s')$ are known for all choices of L_p, m', and s', we can compute $w^i_{i+k}(L_i, L_{i+k}, m, l, s)$. This computation takes at most linear time because only s' depends on n. Therefore, a subproblem can be solved in $O(n^3)$.
Now, we have to check how many subproblems will be generated. Note that the subproblems are determined by the result of $next(.,.)$. Since G^{impl} is outer-planar, the application of $next(.,.)$ to some specific subproblem will always return a new position (a position that has not been considered in any other subproblem). Hence, we get only $n-2$ (sub)problems that can be split again and take $O(n^3)$ time each. (Notice that the initializing subproblems need only $O(n)$ time.) Thus, we can compute our goal in $O(n^4)$.

3.7 The Incorporation of Folding Energy

The designed mRNA sequence obtained so far is maximal similar to a typical SECIS sequence and structure and its encoded amino acid sequence is similar to the original amino acid sequence of the protein where we wish to insert selenocysteine. Thus, it can but need not fold into the target structure since no folding criterion concerning the free energy is considered so far. Hence, the sequence has to be mutated further in order to enhance a second optimization criterion related to the free energy of folding and assuring a minimum similarity, which has to be retained. On the one hand, this problem contains the full inverse RNA folding problem as described in Section 2.4. On the other hand, we cannot consider the inverse RNA folding problem alone since any mutation done to improve the foldability of the sequence can change the similarity calculated by SECISDesign so far. Therefore, we enhance SECISDesign by a specialized inverse RNA folding procedure that takes both criteria into account, the similarity and the foldability of the designed RNA sequence.

In contrast to *multiobjective Pareto optimization*, where two or more conflicting objective functions are optimized simultaneously, SECISDesign optimizes both objective functions one after another, i.e. the similarity is maximized in a first step as described above, which is followed by an optimization of a second optimization criterion dealing with the foldability of the designed sequence. However during the second step, a certain amount of the optimal similarity (found during the first step) is assured to obtain a final sequence that is nearly *Pareto optimal*, i.e. an improvement of one objective function should causes a decline of the other objective function.

A common way of multiobjective Pareto optimization is to use one combined objective function that is a linear combination or a weighted sum of the single functions, respectively. In the case of our problem, this turned out to be complicated due to the incomparableness of both objective functions since their values are highly different in size. Therefore, it is challenging to find meaningful combinations of weights.

Thus, the new variant of SECISDesign is also designed as a two-step approach. During a first step, an mRNA sequence is designed that is optimal concerning the similarities on the nucleotide and the amino acid level. This sequence *can* fold into a given structure but might have a low probability of adopting it. This sequence is the starting point of the second step, an inverse RNA folding approach that tries to optimize an objective function that evaluates the foldability and, nevertheless, assures a minimal sequence similarity. The constraints on the nucleotide as well as on the amino acid level are retained.

3.7.1 Different Objective Functions

Since we want to ensure a minimal similarity in any case, the objective functions differ in the optimization criterion used to evaluate the foldability of the RNA sequence. During SECISDesign, we are offering three different objective functions, which analyze the foldability:

- minimizing the structure distance $d(T, T_S)$ between the target structure T and the minimum free energy structure T_S of the designed sequence S (*mfe-mode*, as described in Section 2.3.2),
- maximizing the probability $P(T|S)$ of sequence S folding into the desired structure T (*p-mode*, as described in Section 2.3.2), and
- minimizing the average number of incorrect paired nucleotides $w(S)$ of the designed sequence S (*nc-mode*, described below).

Average Number of Incorrectly Paired Nucleotides (nc-mode). This approach was introduced in [DLWP04]. The objective function to be minimized is the average number of incorrectly paired nucleotides $w(S)$ over the equilibrium ensemble of secondary structures $\Omega(S)$ of a sequence S. It is defined as follows:

$$w(S) = 3n - \sum_{\substack{1 \leq i \leq 3n \\ 1 \leq j \leq 3n+1}} P_{ij}(S)\overline{T}_{ij} \tag{3.12}$$

where $3n$ is the sequence length of S and P is the base pair probability matrix (computed by the partition function algorithm [McC90]) with entries $P_{ij} \in [0,1]$ representing the probability that S_i and S_j pair. P has an additional $(3n+1)$-th column containing values $P_{i,3n+1} \in [0,1]$ that equal the probability that S_i is unpaired. $\overline{T}$ represents the secondary structure matrix describing the desired structure T where

$$\begin{aligned} \overline{T}_{ij} &= \begin{cases} 1 & \text{if } T \text{ contains base pair } (i,j) \\ 0 & \text{otherwise} \end{cases} & 1 \leq i,j \leq 3n \\ \overline{T}_{i,3n+1} &= \begin{cases} 1 & \text{if position } i \text{ is unpaired in } T \\ 0 & \text{otherwise} \end{cases} & 1 \leq i \leq 3n \end{aligned} \tag{3.13}$$

Thus, the sum of each row of P and $\overline{T}$ is one. For detailed information see [DLWP04].

Besides single execution, the three approaches can be combined with each other to reach some better results. The combined modes are run one after another. If the local search concerning the objective function of the first mode stops, a new search considering the next function is started and initialized with the final sequence of the first mode.

3.7.2 Different Local Search Methods

Due to the facts that firstly the number of valid solutions grows exponentially in the size of the structure and secondly every sequence can adopt different structures, it is not possible to test all solutions to find the global optimum [Hof94]. Thus similar to INFO-RNA, different heuristic local search strategies (not analyzing the complete solution space) are used during SECISDesign. Furthermore, the Vienna RNA Package [SFSH94] is used to evaluate the energies and the foldings. Here, neighbored sequences are defined similar to Definition 2.3.3.

To start the local search, an initial sequence is needed. In our case, we are using the mRNA sequence having maximal similarity resulting from the first step of SECISDesign as described in section 3.6. Having chosen this start sequence, local optima (concerning to the chosen objective function) are found by using one of the following local search strategies.

Adaptive walk (AW). During the strategy of an adaptive walk, the search moves to the *first* found neighbor of the sequence which has 'better' costs according to an objective function. Often, the adaptive walk is also called *fast local search.* The sequence is mutated as soon as a better neighbor is found without checking the other untested neighbors. In our implementation, the order in which the neighbors are examined is chosen randomly. The search terminates if no better neighbor can be found. The adaptive walk is a strategy that is widely used on the inverse folding, e.g. in [Hof94, DLWP04].

Full local search (FLS). The approach of a full local search is similar to the adaptive walk. However, whereas the AW proceeds to the first found better neighbor, the full local search moves to the *best* neighbor of the sequence according to an objective function. The sequence is mutated when all neighbors are examined and a better one is found. Due to the fact that all neighbors are checked, the order in which it is done is not important. If a sequence has several best neighbors, the choice is taken randomly. The search terminates if no better neighbor can be found. To our knowledge, this approach was not used during the inverse RNA folding yet.

Stochastic local search (SLS). The strategy of stochastic local search was already introduced in Section 2.3.2. Briefly, it has a lot in common with the AW. Whereas the latter often gets stuck in local optima (sequences which are better than all their neighbors but not necessarily the globally best solution), the stochastic local search is allowed to move to worse neighbors with a fixed probability p_w to overcome local optima. It terminates after a fixed number of steps and returns the best found solution.

In order to combine the first and the second step of SECISDesign, the question arises, how optional pairs are to be handled during the local search. Since some base pairs of the SECIS element are optional (i.e. advantageous but not necessary), many different structures are allowed during the first step of SECISDesign. In contrast, the inverse RNA folding of the second step needs one fixed structure.

Therefore, the flexible target SECIS structure T (used in the first step) is fixed to T_f after the first step. In T_f, all optional base pairs that are formed at the end of the initializing step of SECISDesign become mandatory. Then, T_f is used during the second step of SECISDesign.

Finally, we get a sequence $S = S_1 ... S_{3n}$, which folds into structure T_f with high probability and, nevertheless, holds at least a user-given minimum similarity to the SECIS constraints C and to the original amino acid sequence A. The whole algorithm is shown schematically in Figure 3.10.

3.7.3 Assurance of a Minimum Similarity

Even though the two algorithmic parts of SECISDesign are run one after another, a certain level of similarity (optimized during the first step) has to be assured while executing the local search during the second step. Therefore during the second step, the similarity has to be within a certain amount of the similarity sim_{start} (the optimal similarity obtained during the first step of SECISDesign). Thus, the minimal allowed similarity is a fixed fraction p_c of the similarity to be compared with. Mutations are allowed only if the local search approach accepts them *and* a minimum similarity is kept, i.e. $sim_{new} > p_c * sim_{start}$. This is done to avoid a final sequence that is totally different from the optimal sequence obtained during the first step of SECISDesign.

3.8 Results

3.8.1 Results without Energy Considerations

During our tests, we used SECISDesign with two different goals; (1) in order to express a mammalian selenoprotein in *E.coli* and (2) to insert selenocysteine in a protein, which naturally has no selenocysteine at the considered position but a cysteine. In both cases, we designed a typical SECIS-element of *E.coli* (*fdhF*) directly after the UGA-position with only a few changes within the amino acid sequence. In *E.coli*, the gene fdhF encodes the selenocysteine-containing enzyme formate dehydrogenase H. According to [HB98, LRGEK98], the SECIS-element of fdhF is still functional if there is an additional codon between the UGA codon and the actual SECIS-element or if the first codon of the SECIS-element is missing. We chose SECIS sequence constraints and structures as shown in Figure 3.11.

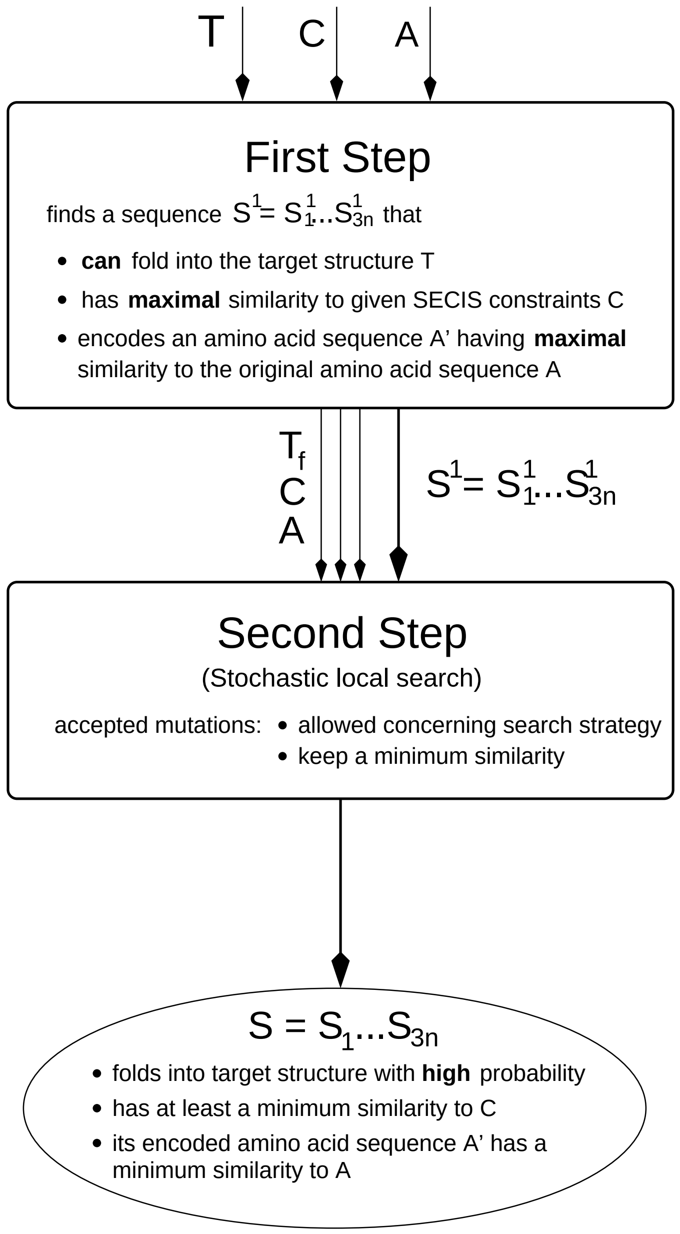

T
C
A
First Step
finds a sequence $S^1 = S^1_1...S^1_{3n}$ that
• **can** fold into the target structure T
• has **maximal** similarity to given SECIS constraints C
• encodes an amino acid sequence A' having **maximal** similarity to the original amino acid sequence A
T_f
C
A
$S^1 = S^1_1...S^1_{3n}$
Second Step
(Stochastic local search)
accepted mutations:
• allowed concerning search strategy
• keep a minimum similarity
$S = S_1...S_{3n}$
• folds into target structure with **high** probability
• has at least a minimum similarity to C
• its encoded amino acid sequence A' has a minimum similarity to A

Figure 3.10: Work flow of SECISDesign including both steps.

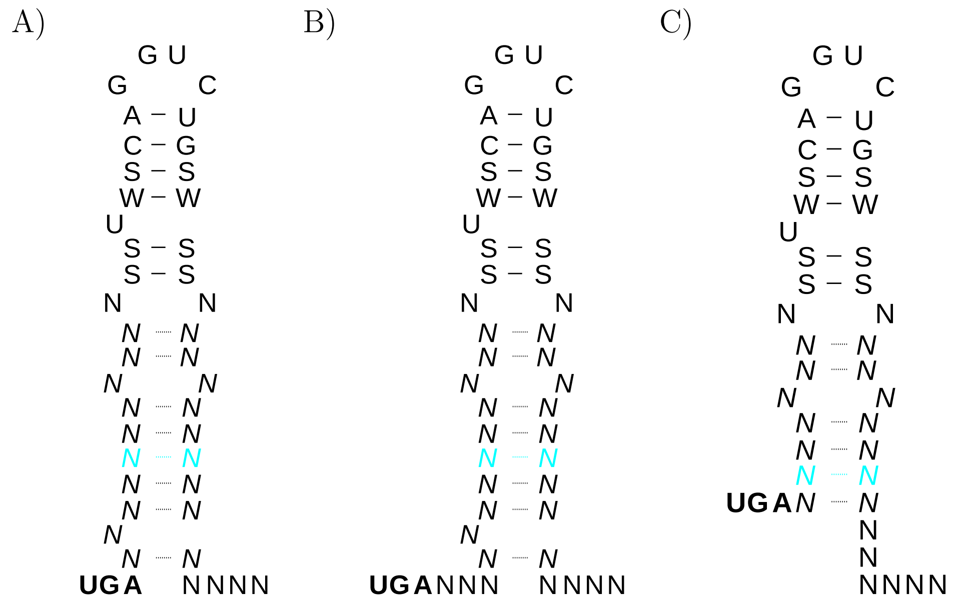


Figure 3.11: SECIS sequence constraints and structure: A) standard, B) an additional codon between UGA and the actual SECIS-element, and C) standard, lacking the first codon. Optional and unfavorable pairs are given in italic type, (optional) G-U pairs are given in cyan. Constraints on the nucleotides are given in IUPAC symbols (see Table 2.3).

Expressing Mammalian Selenoproteins in *E.coli*

The mammalian methionine sulfoxide reductase B (MsrB) has been found to be a selenoprotein, which can reduce free and protein-incorporated methionine sulfoxide to methionine [BNM02]. It has been identified as a selenoprotein only in mammals. To identify and analyze its active site and the effect of the selenocysteine on its function, large amounts of MsrB are needed. To allow an MsrB expression in *E.coli*, we designed a SECIS-element of *E.coli* directly after the UGA codon coding for selenocysteine at amino acid position 95.
Similar tests, but using an handcrafted design, have already been done by Bar-Noy and Moskovitz [BNM02]. They found an RNA sequence that results in an amino acid sequence with six amino acids changed and a similarity of 35 (see Table 3.4) when applying the widely used substitution matrix *BLOSUM 62* (BLOck SUbstitution Matrix) to score the similarity of one amino acid to another. Using SECISDesign, we obtained better sequences with higher similarities to the original amino acid sequence when designing the SECIS-elements shown in Figure 3.11. The results are shown in Table 3.4.

Furthermore, we applied SECISDesign to all human selenoproteins, which include at least one selenocysteine that is encodes at least 11 codons upstream of the real stop-codon. This is due to the fact that only in the mRNAs coding for these proteins, the SECIS-element is located largely in the coding sequence of the protein. For all remaining human selenoproteins, only a small part (up to five codons) of the designed SECIS-element is located in the coding sequence, while most codons are in the 3' UTR. Thus, the problem is less complicated since there are many degrees of freedom due to the part of the SECIS-element that does not code for any amino acid. A list of know human selenoproteins is given in Table 3.5.
We found modifications of all tested human selenoproteins that have high similarity to both the original amino acid sequence and the SECIS-element. Since a SECIS-element does not fit in every coding mRNA equally good, we used the three variants of the SECIS-element shown in Figure 3.11. In some cases, all three variants of the SECIS-element can be incorporated successfully, while in other cases, only one version gives satisfying results (see Table 3.13). Thus, it is always recommended to test several variants of a possible SECIS-element. Then, the user can either choose among several good results or has at least a chance to find one acceptable result.
All tested human selenoproteins, in which the standard SECIS-element fits best, and their modified versions including the standard SECIS-element in the coding mRNA are shown in Table 3.6. Analogously, the human selenoproteins, in which the SECIS-element having an additional codon or the SECIS-element missing the first codon fits best, and their modified versions are shown in Tables 3.7 and 3.8,

Bar-Noy *et al.*	mouse MsrB	... I F S S S L K F V P K G K E ...
	AA_seq [BNM02]	... I F S T V A G L H P K G K E ...
	mRNA_seq	... AUAUUUAGCACGGUUGCAGGUCUGCACCCUAAAGGCAAAGAA
	mRNA_struct	... .((.((((....))))))..............
	similarity	35
A)	mouse MsrB	... I F S S S L K F V P K G K E ...
	AA_seq	... I F S S L P G L V P K G K E ...
	mRNA_seq	... AUUUUCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAAAAGAA
	mRNA_struct	... ..(((((.((.((.((((....)))))).)).))))).....
	similarity	43
	optional pairs	9 / 9
B)	mouse MsrB	... I F S S S L K - F V P K G K E ...
	AA_seq	... I F S S S L P G L V P K G K E ...
	mRNA_seq	... AUAUUUUCCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAAAAGAA
	mRNA_struct	... ...(.(((((.((.((.((((....)))))).)).))))))....
	similarity	47
	optional pairs	9 / 9
C)	mouse MsrB	... I F S S S L K F V P K G K E ...
	AA_seq	... I F S L P - G L V P K G K E ...
	mRNA_seq	... AUCUUUUCGCUACCAGGUCUGGUGCCAAAAGGUAAAGAA
	mRNA_struct	... (((((((.((.((((....)))))).)))))))......
	similarity	36
	optional pairs	7 / 7

Table 3.4: Results for MsrB, reported in [BNM02] and using SECISDesign for the design of A) the standard SECIS-element shown in Figure 3.11A, B) the SECIS-element of Figure 3.11B, and C) the standard SECIS-element missing the first codon shown in Figure 3.11C. IP is set to -10, DP to -5. Changed positions are given in red. Sequences start after the positions where selenocysteine is inserted. AA_seq represents the amino acid sequence encoded by the designed RNA sequence, which is named mRNA_seq. The secondary structure, which the designed RNA sequence adopts, is given by mRNA_struct. Similarities of the designed amino acid sequences to the original one are calculated using BLOSUM 62. Furthermore, the number of realized optional (G-U) pairs and not realized unfavorable pairs is given additionally to their maximally realizable number.

Name	Sec location in the protein	Protein length
15kDa	93	162
IOD1	126	249
IOD2	133	265
IOD3	144	278
GPX1	47	201
GPX2	40	190
GPX3	73	226
GPX4	73	197
GPX6	73	221
MsrB	95	116
SelH	44	122
SelI	387	397
SelK	92	94
SelM	48	145
SelN	428	556
SelO	667	669
SelP	59, 300, 318, 330, 345, 352, 367, 369, 376, 378	381
SelS	188	189
SelT	17	163
SelV	273	346
SelW	12	86
SPS2	60	448
TXN1	497	497
TXN2	522	524

Table 3.5: List of human selenoproteins. Basic information is taken from Kryukov *et al.* [KCN+03] and modification are made concerning to the EMBL nucleotide sequence database [KAA+07]. Proteins given in light gray are not analyzed using SECISDesign since their included selenocysteines are at most six amino acids upstream of the real stop-codons and thus, the most part of the SECIS-element is located in the UTR and not coding for any amino acid.

respectively. Based on these modifications, the coding mRNA is able to form the required hairpin structure.

Insertion of a New Selenocysteine

We further examined a subunit of ribonucleoside-diphosphate reductase (RIR1) of *E.coli*. Naturally, it has no selenocysteine at position 439 but a cysteine. Furthermore, the glutamic acid at position 441 is highly conserved and must not be changed. We designed the SECIS-elements shown in Figure 3.11 directly after the modified codon (from UGC to UGA) in order to insert a selenocysteine. The results are shown in Table 3.9.

3.8.2 Results Involving Folding Energies

Incorporating the second step of SECISDesign, we tried to improve the results obtained in Section 3.8.1 concerning the probability of folding into the desired SECIS structure but nevertheless to ensure a certain amount of similarity.
For that purpose, we tested each combination of the parameters shown in Table 3.10. Since most search strategies are not deterministic, we have examined each combination 100 times (except full local search) to get information about the average quality of each approach (data not shown).
It is difficult to come to a general conclusion about the best parametric combination. However, it can be said that in nearly all cases the full local search produces the best results concerning both similarity and folding probability. Furthermore, combinations of the modes provide better results than their single execution. Concerning the relative factor p_c, no general recommendation is possible. However, in most cases, higher values assure a better result. When $p_c = 50\%$ or $p_c = 60\%$, the results show a very high probability of folding into the target structure but a quite low similarity which is not useful for our intention but shows the good performance of the inverse folding procedure itself. Some of the best results for MsrB and RIR1 are given in more detail in Tables 3.11 and 3.12, respectively.
Compared to [BNM02], the application of SECISDesign leads to better results concerning the quality of the SECIS-element (since the bases of the lower part of the stem are paired), the probability of folding into the target structure, and often, also concerning the similarity score.
As already mentioned in Section 3.8.1, not every SECIS-element fits in every coding mRNA sequence equally good. Thus, several versions of a SECIS-element should always be tested. The results given in Table 3.13 show that, in case of human SelP, all three variants of the *E.coli* SECIS-element fdhF fit relatively good in the coding mRNA and result in only a few changed amino acids. In contrast, the

Incorporating the SECIS-element given in Figure 3.11A:

15kDa(O60613)	… K L G R F P Q V Q A F V R S …
	… K L G R L P G L E G F V R S …
IOD1(P49895)	… P S F M F K - F D Q F K R L …
	… P S F M V P G L D Q F K R L …
GPX2(P18283)	… G T T T R D F T Q L N E L Q …
	… G T T T L A G L Q L N E L Q …
GPX6(P59796)	… G L A A Q Y P E L N A L Q E E …
	… G L A A - V P G L D A L Q E E …
SelM(Q8WWX9)	… Q L N R L K E V K A F V T Q …
	… Q L N R L A G L Q G F V T Q …
SelP$_{300}$(P49908)	… C C H C R H L I F E K T G S …
	… C C H C L A G L L E K T G S …
SelP$_{367}$(P49908)	… R C K N Q A K K C E C P S N …
	… R C K N L A G L L E C P S N …
SelP$_{369}$(P49908)	… K N Q A K K - C E C P S N * …
	… K N Q A L P G L E A P S N * …

Table 3.6: Results for all human selenoproteins that showed the best similarity after incorporating the standard SECIS-element shown in Figure 3.11A. We set IP to -10 and DP to -5. Changed positions are given in red. Proteins names and their accession numbers are given. Changed sequences are given below the natural ones. Since selenoprotein SelP includes more than one selenocysteine, the referred selenocysteine positions are indicated in the subscript of its name. In case of the selenocysteine at position 367 in SelP, the selenocysteine at position 369 is substituted by a cysteine. All sequences start after the positions where the selenocysteine is inserted. Positions corresponding to nucleotides in the UTR are indicated by *.

Incorporating the SECIS-element given in Figure 3.11B:	
GPX4(P36969)	... G K T E V N Y T Q L V D L H A R ...
	... G K T E - N L P G L V D L H A R ...
MsrB(Q9NZV6)	... I F S S S L K - F V P K G K E ...
	... I F S S S L P G L V P K G K E ...
SelH(Q8IZQ5)	... R V Y G R N A A A L S Q A L R L ...
	... R V Y G R - L A G L Q Q A L R L ...
SelI(Q9C0D9)	... L G M E E K N I G L * * * * * * ...
	... L G M E E L - A G L * * * * * * ...
SelN(Q9NZV5)	... G S G R T L R E T V L E S S P ...
	... G S G R T L P G L V L E S S P ...
SelP_{59}(P49908)	... Y L C I I E A S K L E D L R V ...
	... Y L C I I V A G L L E D L R V ...
SelP_{318}(P49908)	... Q C K E N L P S L C S C Q G L ...
	... Q C K E N L P G L V A C Q G L ...
SelP_{330}(P49908)	... Q G L R A E E N I T E S C Q C ...
	... Q G L R A V P G L V E S C Q C ...
SPS2(Q99611)	... G C K V P Q E A L L K L L A G ...
	... G C K V P L A G L L E L L A G ...

Table 3.7: Results for all human selenoproteins that showed the best similarity after incorporating the SECIS-element of Figure 3.11B, which has an additional codon between the UGA and the actual SECIS-element. IP was set to -10 and DP to -5. Changed positions are given in red. Proteins names and their accession numbers are given. Changed sequences are given below the natural ones. Since selenoprotein SelP includes more than one selenocysteine, the referred selenocysteine positions are indicated in the subscript of its name. All sequences start after the positions where the selenocysteine is inserted. Positions corresponding to nucleotides in the UTR are indicated by *.

Incorporating the SECIS-element given in Figure 3.11C:	
IOD2(Q92813)	... P P F T S Q L P A F R K L ...
	... P P F L A G L Q A F R K L ...
IOD3(P55073)	... P P F M A R M S A F Q R L ...
	... P P F L A G L Q A F Q R L ...
GPX1(P07203)	... G T T V R D Y T Q M N E L ...
	... G T T V A G L L Q M N E L ...
GPX3(P22352)	... G L T G Q Y I E L N A L Q ...
	... G L T L P G L E L N A L Q ...
SelP$_{345}$(P49908)	... R L P P A A C Q I S Q Q L ...
	... R L P L A G L Q V S Q Q L ...
SelP$_{352}$(P49908)	... Q I S Q Q - L I P T E A S ...
	... Q I S L P G L V P T E A S ...
SelT(Q9NZJ3)	... G Y R R V - F E E Y M R V ...
	... G Y R L P G L E E Y M R V ...
SelV(P59797)	... S Y S L R Y I L L K K S L ...
	... S Y S L A G L L L K K S L ...
SelW(O15532)	... G Y K S K Y L Q L K K K L ...
	... G Y K L A G L Q L K K K L ...

Table 3.8: Results for all human selenoproteins that showed the best similarity after incorporating the SECIS-element missing the first codon shown in Figure 3.11C. We set IP to -10 and DP to -5. Changed positions are given in red. Proteins names and their accession numbers are given. Changed sequences are given below the natural ones. Since selenoprotein SelP includes more than one selenocysteine, the referred selenocysteine positions are indicated in the subscript of its name. All sequences start after the positions where the selenocysteine is inserted. In case of the selenocysteine at position 345 in SelP, the selenocysteine at position 352 is substituted by a cysteine.

A)		
	E.coli RIR1	... L E I A L P T K P L N D V N ...
	AA_seq	... L E I A L P G L E L N D V N ...
	mRNA_seq	... CUCGAAAUUGCGCUUCCAGGUCUGGAGCUCAAUGACGUAAAC
	mRNA_struct	... ..((..((((.((.((((....)))))).)))).)).....
	similarity	49
	optional pairs	5 / 9
B)		
	E.coli RIR1	... L E I A L P T K P L N D V N D E ...
	AA_seq	... L E I - L P V A G L H D V N D E ...
	mRNA_seq	... CUAGAAAUUCUCCCCGUUGCAGGUCUGCACGACGUGAAUGACGAA
	mRNA_struct	... .((((.(..((.((((....))))))..).))))......
	similarity	43
	optional pairs	6 / 9
C)		
	E.coli RIR1	... L E I A L P T K P L N D V N ...
	AA_seq	... L E - A L P G L E L N D V N ...
	mRNA_seq	... CUCGAAGCGCUUCCAGGUCUGGAGCUCAACGACGUAAAC
	mRNA_struct	... .(((..(.((.((((....)))))).)..))).......
	similarity	35
	optional pairs	5 / 7

Table 3.9: Results for RIR1 using SECISDesign for the design of A) the standard SECIS-element shown in Figure 3.11A, B) the SECIS-element of Figure 3.11B, and C) the standard SECIS-element missing the first codon shown in Figure 3.11C. IP was set to -10, DP to -5, changed positions are given in red. Sequences start after the positions where selenocysteine is to be inserted. AA_seq represents the amino acid sequence encoded by the designed RNA sequence, which is named mRNA_seq. The secondary structure, which the designed RNA sequence adopts, is given by mRNA_struct. Similarities of the designed amino acid sequences to the original one are calculated using BLOSUM 62. Furthermore, the number of realized optional (G-U) pairs and not realized unfavorable pairs is given additionally to their maximally realizable number.

Objective function	Search strategy	p_c [%]
mfe-mode	AW	50
p-mode	FLS	60
nc-mode	SLS	70
mfe-mode→p-mode		80
mfe-mode→nc-mode		90
nc-mode→p-mode		
mfe-mode→nc-mode→p-mode		

Table 3.10: Tested parameters of SECISDesign. If the mfe-mode is used, it is executed first since it proceeds fastest but worst.

insertion of the three variants of the SECIS-element in the coding mRNA of human GPX3 shows only in case of the SECIS-element missing the first codon (shown in Figure 3.11C) a satisfying result.
Furthermore, the results in Table 3.13 show that the second step of SECISDesign is highly important and needed in most cases. Again, it is useful to have several variations of the SECIS-element to be tested.

3.9 Discussion

We have introduced a successful new approach to the design of SECIS-elements in the coding region of a protein, which is called SECISDesign.
It consists of two major steps. During the first step, an RNA sequence is designed that is optimal concerning its similarity to a SECIS constraints sequence and concerning the similarity its encoded amino acid sequence has compared to the original amino acid sequence of the protein. Furthermore, the typical hairpin-stem structure of a SECIS-element can be adopted. Since, during the first step, free energy of folding is not taken into account, a second step is needed. There, the designed sequence of the first step is mutated further to enhance its folding ability and, nevertheless, ensure at least a given minimal similarity (which had been optimized during the first step). The final sequence of the first step is an excellent starting point for the subsequent local search of the second step. Here, several objective function concerning the foldability as well as several local search methods are available in SECISDesign.

	AA sequences[a]	Sim.[b]	mRNA sequences and structures	Prob.[c]
MsrB	`IFSSSLKFVPKGKE`			

MsrB^{BaNo}	`IFSTVAGLHPKGKE`	35	`AUAUUUAGCACGGUUGCAGGUCUGCACCCUAAAGGCAAAGAA`
			`...........((.((((....))))))..............`
				0.01

MsrB^{1}	`IFSSLPGLVPKGKE`	43	`AUUUUCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAAAAGAA`
			`..(((((.((.((.((((....)))))).)).))))).....`
				0.04

MsrB^{mfe}	`IFSSLPGLVPKGTE`	37	`AUUUUCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAACAGAA`
			`..(((((.((.((.((((....)))))).)).))))).....`
				0.23

MsrB$^{mfe}_{nc}$	`IFSSLPGLVPQGAE`	33	`AUCUUCUCGUCGCUACCAGGUCUGGUGCCACAAGGAGCCGAA`
			`..(((((.((.((.((((....)))))).)).))))).....`
				0.75

MsrB	original amino acid sequence of MsrB in mouse
MsrB^{BaNo}	sequence given in [BNM02]
MsrB^{1}	using only the first step of SECISDesign (no optimization of the stability, [BB04])
MsrB^{mfe}	using mfe-mode and FLS during the second step of SECISDesign
MsrB$^{mfe}_{nc}$	using mfe-mode and nc-mode afterwards and FLS during the second step of SECISDesign (and default values for other parameters)

[a] amino acid sequences encoded by the designed RNA sequences
[b] similarity of the designed amino acid sequence to the original sequence using BLOSUM 62
[c] folding probability

Table 3.11: Results for the design of fdhF as shown in Figure 3.11A in mouse MsrB when using SECISDesign. Changed positions are given in red. All sequences start after the positions where selenocysteine is inserted.

	AA sequences[a]	Sim.[b]	mRNA sequences and structures	Prob.[c]
RIR1	LEIALPTKPLNDVN			
RIR1^1	LEIALPGLELNDVN	49	CUCGAAAUUGCGCUUCCAGGUCUGGAGCUCAAUGACGUAAAC ..((..((((.((.((((....)))))).))))..)).....	0.30
RIR1^{mfe}	LEIALPGLVLNDVN	48	CUCGAAAUUGCGCUACCAGGUCUGGUGCUCAAUGACGUAAAC ..((..((((.((.((((....)))))).))))..)).....	0.61
RIR1$^{mfe}_{nc}$	LEIALPGLVLNDLN	45	CUGGAAAUUGCGCUACCAGGUCUGGUGCUCAAUGACCUUAAC ..((..((((.((.((((....)))))).))))..)).....	0.79

RIR1	original amino acid sequence of RIR1 in *E.coli*
RIR1^1	using only the first step of SECISDesign (no optimization of the stability, [BB04])
RIR1^{mfe}	using mfe-mode, FLS, and $p_c = 90\%$ during the second step of SECISDesign
$RIR1^{mfe}_{nc}$	using mfe-mode and nc-mode afterwards, FLS, and $p_c = 90\%$ during the second step of SECISDesign (and default values for other parameters)

[a] amino acid sequences encoded by the designed RNA sequences
[b] similarity of the designed amino acid sequence to the original sequence using BLOSUM 62
[c] folding probability

Table 3.12: Results for the design of fdhF as shown in Figure 3.11A in *E.coli* RIR1 when using SECISDesign. Changed positions are given in red. All sequences start after the positions where selenocysteine is inserted.

				Sim.[3]	Prob.[4]
1)	SelP$_{300}$(P49908)				
	(A)	original aa seq.	... C C H C R H L I F E K T G S ...		
		designed aa seq[1]	... C C H C L A G L L E K T G S ...	54	0.02808
		designed aa seq[2]	... C C H H V A G L L E K T E S ...	33	0.80059
	(B)	original aa seq	... C C H C R H L I F E K T G S A ...		
		designed aa seq[1]	... C C H C R L P G L E R T G S A ...	56	0.00051
		designed aa seq[2]	... Y Y H C R L P G L E R T G S A ...	34	0.24811
	(C)	original aa seq	... C C H C R H L I F E K T G ...		
		designed aa seq[1]	... C C H L A G L L A E K T G ...	47	0.00009
		designed aa seq[2]	... C H H V A G L L E E R T G ...	31	0.42835
2)	GPX3(P22352)				
	(A)	original aa seq	... G L T G Q Y I E L N A L Q E E L A P ...		
		designed aa seq[1]	... G - T G - Y L - A - G L Q E E L A P ...	39	0.00031
		designed aa seq[2]	... D - P G - Y V - A - G L H E E Q A P ...	16	0.24698
	(B)	original aa seq	... G L T G Q Y I E L N A L Q E E L A P ...		
		designed aa seq[1]	... G - T G Q Y L - A - G L Q E E L A P ...	49	0.00680
		designed aa seq[2]	... K - T G Q H V - A - G L L E E L Q P ...	25	0.85494
	(C)	original aa seq	... G L T G Q Y I E L N A L Q ...		
		designed aa seq[1]	... G L T L P G L E L N A L Q ...	37	0.37718
		designed aa seq[2]	... G L T V A G L L L D A I Q ...	23	0.86739

[1] amino acid sequence after the first step of SECISDesign
[2] amino acid sequence after the second step of SECISDesign (using p-mode and FLS)

[3] similarity to the original amino acid sequence
[4] probability that the coding mRNA folds into the SECIS structure

Table 3.13: Selected results for human selenoproteins, demonstrating the difference in quality when inserting different variants of a SECIS-element. While in case of SelP$_{300}$ all SECIS-element variations give relatively good results, only one version of the SECIS-element gives a satisfying result in case of GPX3. We designed (A) the standard SECIS-element shown in Figure 3.11A, (B) the SECIS-element of Figure 3.11B, and (C) the SECIS-element missing the first codon shown in Figure 3.11C. We set IP to -10 and DP to -5. Changed positions are given in red. Proteins names and their accession numbers are given. Changed sequences are given below the natural ones. Since the selenoprotein SelP includes more than one selenocysteine, the referred selenocysteine position is indicated in the subscript of its name. All sequences start after the positions where the selenocysteine is inserted.

It should be said that it is not possible to execute both algorithmic parts simultaneously since the first step is a dynamic programming procedure where optimal partial solutions are part of the complete optimal one. This is not the case if we consider the free energy. Although substructures contribute additively to the energy, an optimal substructure is not guaranteed to be optimal if we consider the full sequence [Hof94].

To test the performance of SECISDesign, we designed a typical SECIS-element of *E.coli* (*fdhF*) directly after the UGA-position coding for selenocysteine with only a few changes within the amino acid sequence to allow its expression in *E.coli.* This design was applied with two different goals: (i) in order to express a mammalian selenoprotein in *E.coli* and (ii) to insert selenocysteine in a protein that naturally has no selenocysteine at the considered position but a cysteine.
For most tested proteins, we obtained good results, i.e. stable SECIS-elements as well as only few changes in the amino acid sequences of the proteins. It should be noted that a unique best combination of all parameters can not be given, even if the full local search was the most successful search in the majority of the cases.
To conclude, SECISDesign is a successful tool for the design of SECIS-elements inside the coding region, which by far outperforms handcrafted design.

Chapter 4

Web Services

To provide access to our programs INFO-RNA and SECISDesign to a wide community of scientists, we are offering a web service for each of them. It allows to use INFO-RNA and SECISDesign without compiling source codes or having an executable version on the local computer.

4.1 INFO-RNA Web Server

The INFO-RNA web server allows biologists to design RNA sequences, which fold into a user given secondary structure, in an automatic manner. Furthermore, constraints on the nucleotide sequence can be specified as well as violations of them can be allowed by the user. The INFO-RNA web server is clearly and intuitively arranged and easy to use. The procedure is fast, as most applications are completed within seconds. As shown in Section 2.5, INFO-RNA leads to better results and is faster than other existing tools. It was first introduced in [BB07] and is available at

http://www.bioinf.uni-freiburg.de/Software/INFO-RNA/

4.1.1 Functionality and Usage

In order to obtain an RNA sequence folding into a target structure and satisfying some sequence constraints, structure and sequence constraints have to be given. Thereto, the user has to specify the secondary structure in dot-bracket notation and the sequence constraints have to be entered in IUPAC symbols (see Table 2.3). Additionally, the user can choose some positions where the constraints are allowed to be violated during the local search. These have to be marked with a '+' next to the constraint of the position. In contrast, strict constraints have to be marked

with a '-'. Furthermore, the maximal number of positions where the constraints are allowed to be violated in the final sequence can be specified.
The user has to fix some parameters used during the stochastic local search, e.g. the search strategy of either only minimizing the structure distance or additionally maximizing the folding probability as well as the search order of the sequence positions.
Finally, the user can choose whether the results are shown on the web page or sent via email. To facilitate the use of INFO-RNA, we set recommended values by default. For all options, a comprehensive help and detailed examples are given. Figure 4.1 shows the input page of the INFO-RNA web server. The input example in this figure is used to design an iron responsive element (IRE) with fixed bases in the interior and hairpin loop and a maximum of two violated constraints at three possible sequence position. This design was already described in Section 2.5.3.

4.1.2 Results and Output

Figure 4.2 shows the output of the computation requested in Figure 4.1. First, the input data are summarized. Below, the designed sequence is shown including information about its minimum free energy (mfe) structure, its free energy, and its folding probability. Furthermore, it is shown whether the mfe structure equals the target structure. Additionally, the user can download the results as a FASTA, CT, and RNAML file.

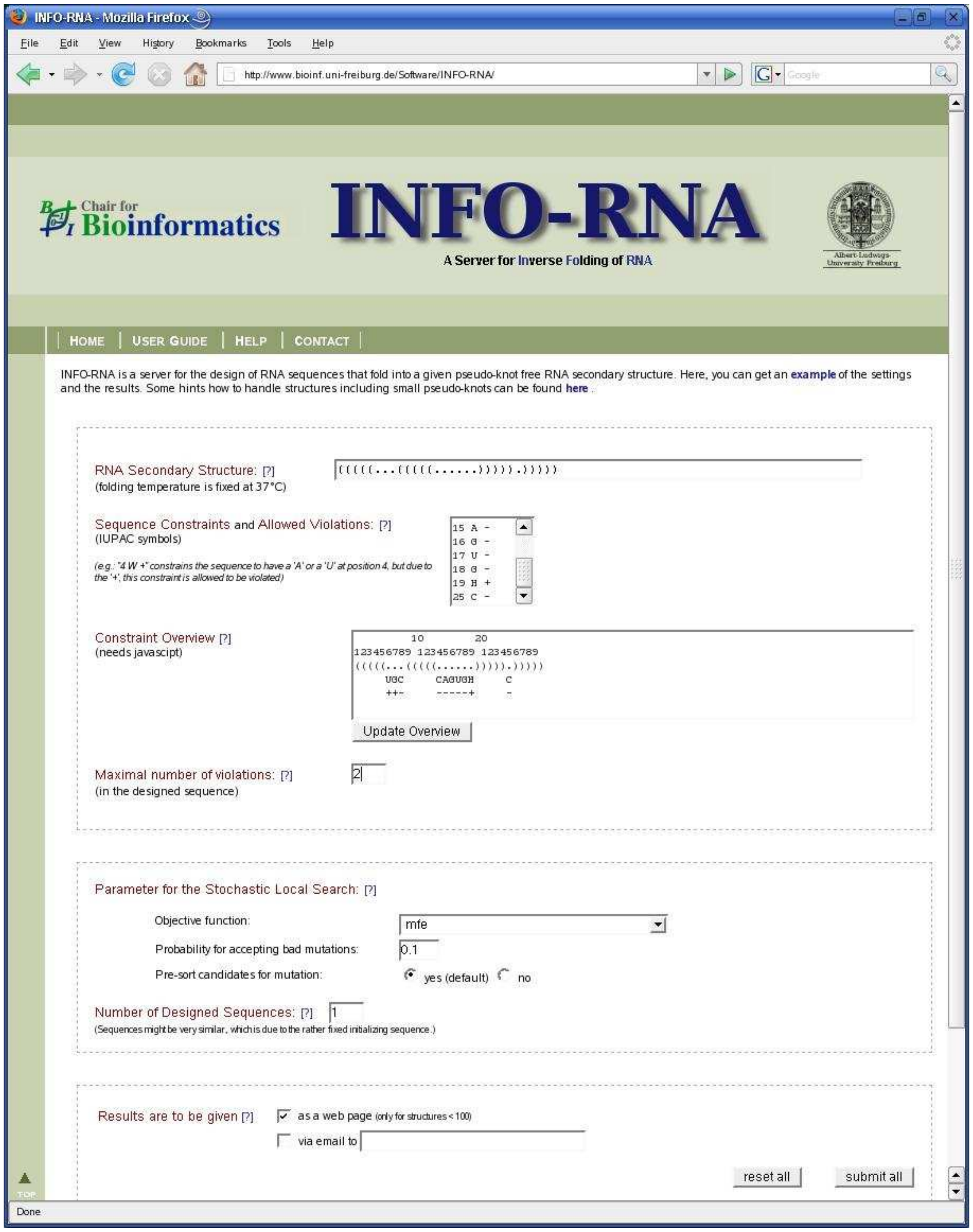
4.2 SECISDesign Web Server

SECISDesign is a server for the design of SECIS-elements and arbitrary, pseudoknot-free RNA-elements within the coding sequence of an mRNA. The element has to satisfy both structure and sequence constraints. At the same time, a certain amino acid similarity to the original protein is kept. The designed sequence can be used e.g. for recombinant expression of selenoproteins in *E.coli*. The web server was first introduced in [BWB05] and is available at

http://www.bioinf.uni-freiburg.de/Software/SECISDesign/

4.2.1 Functionality and Usage

In order to obtain an mRNA sequence coding for a selenocysteine containing protein, the SECISDesign server needs the amino acid sequence and the position, where


INFO-RNA - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.bioinf.uni-freiburg.de/Software/INFO-RNA/
Chair for Bioinformatics
INFO-RNA
A Server for Inverse Folding of RNA
Albert-Ludwigs-University Freiburg
HOME | USER GUIDE | HELP | CONTACT
TOP
Done

INFO-RNA is a server for the design of RNA sequences that fold into a given pseudo-knot free RNA secondary structure. Here, you can get an **example** of the settings and the results. Some hints how to handle structures including small pseudo-knots can be found **here** .

RNA Secondary Structure: [?]
(folding temperature is fixed at 37°C)

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

Sequence Constraints and Allowed Violations: [?]
(IUPAC symbols)

(e.g.: "4 W +" constrains the sequence to have a 'A' or a 'U' at position 4, but due to the '+', this constraint is allowed to be violated)

```
15 A -
16 G -
17 U -
18 G -
19 H +
25 C -
```

Constraint Overview [?]
(needs javascipt)

```
         10        20
123456789 123456789 123456789
(((((...(((((......))))).)))))
      UGC     CAGUGH     C
      ++-     -----+     -
```

Update Overview

Maximal number of violations: [?] 2
(in the designed sequence)

Parameter for the Stochastic Local Search: [?]

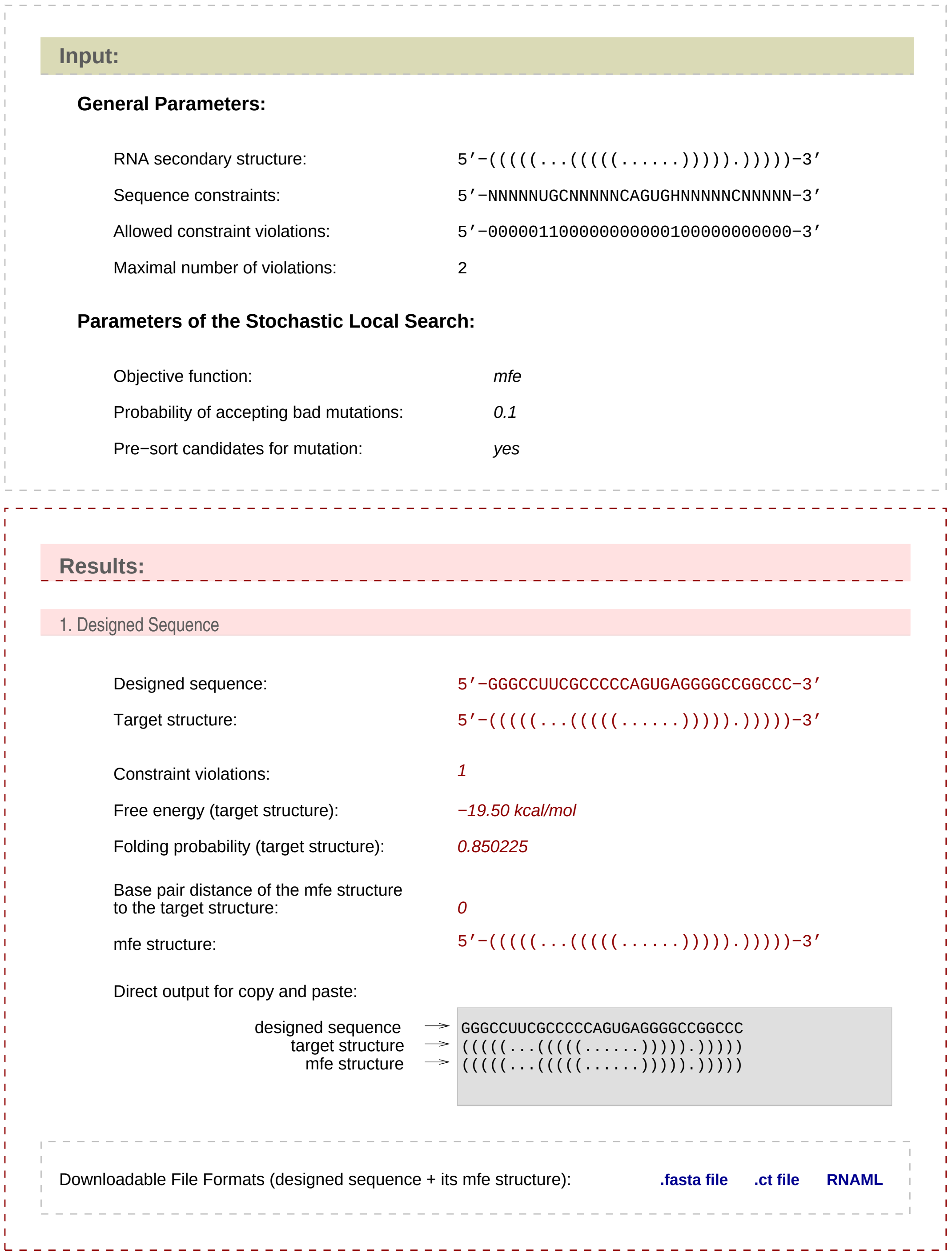
Objective function:	mfe
Probability for accepting bad mutations:	0.1
Pre-sort candidates for mutation:	[✓ yes (default)] no

Number of Designed Sequences: [?] 1
(Sequences might be very similar, which is due to the rather fixed initializing sequence.)

Results are to be given [?]
[✓] as a web page (only for structures < 100)
[ ] via email to

reset all   submit all

Figure 4.1: The figure shows the input page of the INFO-RNA web server, when designing an iron responsive element (IRE) with fixed bases in the interior and hairpin loop and a maximum of two violated constraints at three possible sequence position.



Input:

General Parameters:

RNA secondary structure:	5′−(((((...(((((......))))).)))))−3′
Sequence constraints:	5′−NNNNNUGCNNNNNCAGUGHNNNNNCNNNNN−3′
Allowed constraint violations:	5′−000001100000000000100000000000−3′
Maximal number of violations:	2

Parameters of the Stochastic Local Search:

Objective function:	*mfe*
Probability of accepting bad mutations:	*0.1*
Pre−sort candidates for mutation:	*yes*

Results:

1. Designed Sequence

Designed sequence:	5′−GGGCCUUCGCCCCCAGUGAGGGGCCGGCCC−3′
Target structure:	5′−(((((...(((((......))))).)))))−3′
Constraint violations:	*1*
Free energy (target structure):	*−19.50 kcal/mol*
Folding probability (target structure):	*0.850225*
Base pair distance of the mfe structure to the target structure:	*0*
mfe structure:	5′−(((((...(((((......))))).)))))−3′

Direct output for copy and paste:

```
designed sequence  →  GGGCCUUCGCCCCCAGUGAGGGGCCGGCCC
 target structure  →  (((((...(((((......))))).)))))
    mfe structure  →  (((((...(((((......))))).)))))
```

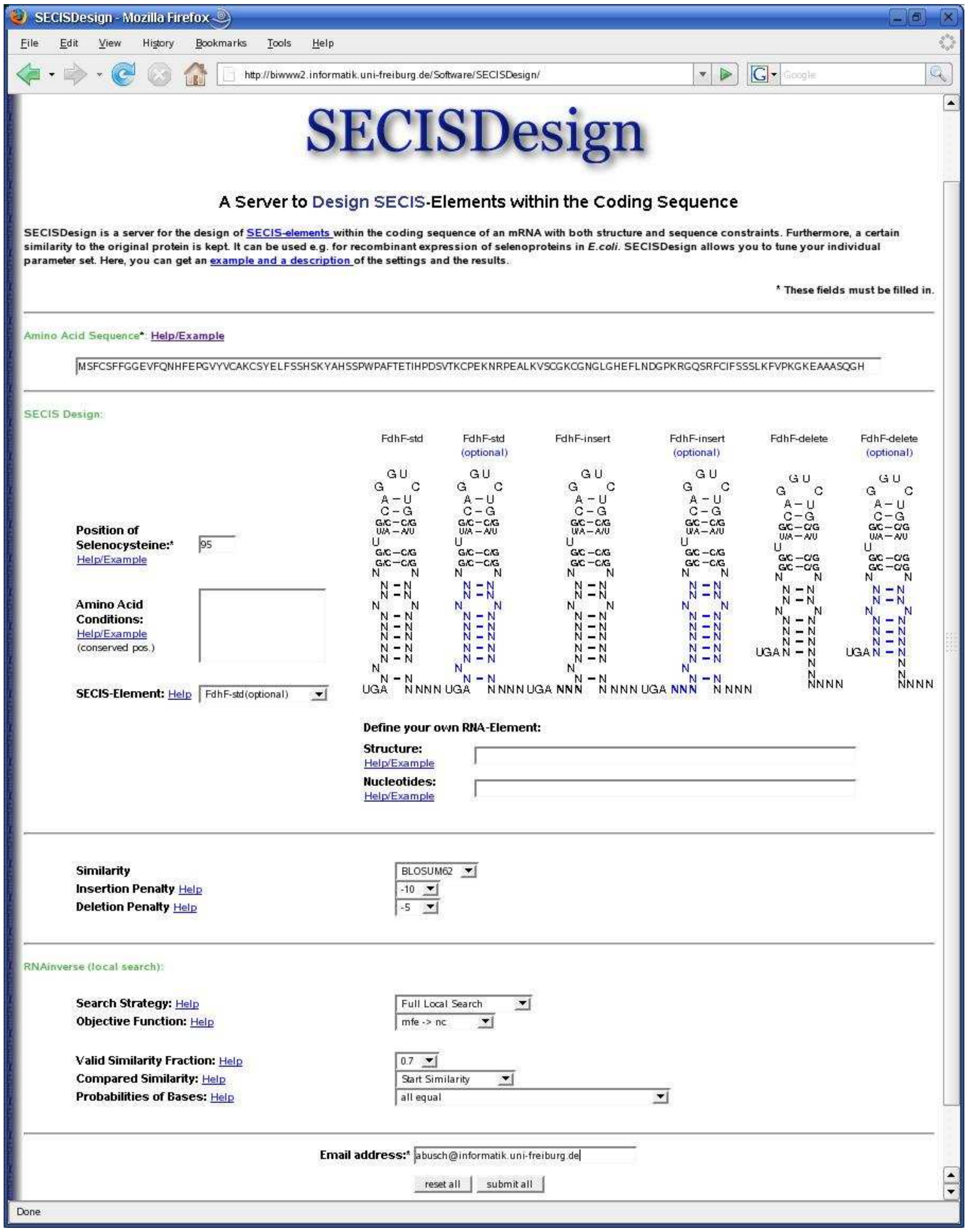
Downloadable File Formats (designed sequence + its mfe structure): **.fasta file** **.ct file** **RNAML**

Figure 4.2: The figure shows the INFO-RNA web server output of the computation requested in Figure 4.1.

the selenocysteine should be inserted. In addition, the user can give information about positions where changes are forbidden or restricted to certain amino acids.
Moreover, the target element can be selected from a list of SECIS-elements of *E.coli* already introduced in Figure 3.11. Although the server is tailored for SECIS-elements, the method is not restricted to the motifs of the list. The user can define new and even unrelated elements by specifying their structure and nucleotide sequence constraints. The latter have to be given in IUPAC symbols shown in Table 2.3. Additionally, the user can choose the substitution matrix that is used to evaluate the similarity of amino acids and the penalties that are added if an amino acid is inserted or deleted.
Furthermore, one can select several parameters of the local search, e.g. the search strategy itself and the evaluation of the foldability. To facilitate the use of SECISDesign, we set recommended values by default. Since the computation usually takes only about one minute, the user can test several parameters to find a solution that fits his requirements best.
Finally, the user has to enter an email address where the results are sent to. For all options, a comprehensive help and detailed examples are given. Figure 4.3 shows the input page of the SECISDesign web server.

4.2.2 Results and Output

Figure 4.4 shows the output of the computation requested in Figure 4.3. On the left side, the input data are summarized. On the right, the target SECIS structure is shown. Furthermore, the designed mRNA sequence without optimizing its stability, i.e. the best sequence after the first step of SECISDesign, as well as the final mRNA sequence after optimizing its stability are given. For both sequences, the probabilities of folding into the target structure are shown.
If the target structure is not the structure of minimum free energy of the designed sequence, its mfe structure is given as well. If it is given in green, it is valid as well but might have less adopted optional pairs. If it is given in red, the mfe structure is not valid concerning the wanted structure. The user might decide whether this structure of minimum free energy fits his requirements anyway. The folding probability is also provided.
Below the designed RNA sequences, the translated amino acid sequences are given. Here, mutated positions are given in blue. Last but not least, a comprehensive help and a detailed example is given.


SECISDesign - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://biwww2.informatik.uni-freiburg.de/Software/SECISDesign/
Done

SECISDesign

A Server to Design SECIS-Elements within the Coding Sequence

SECISDesign is a server for the design of SECIS-elements within the coding sequence of an mRNA with both structure and sequence constraints. Furthermore, a certain similarity to the original protein is kept. It can be used e.g. for recombinant expression of selenoproteins in *E.coli*. SECISDesign allows you to tune your individual parameter set. Here, you can get an example and a description of the settings and the results.

* These fields must be filled in.

Amino Acid Sequence*: Help/Example

MSFCSFFGGEVFQNHFEPGVYVCAKCSYELFSSHSKYAHSSPWPAFTETIHPDSVTKCPEKNRPEALKVSCGKCGNGLGHEFLNDGPKRGQSRFCIFSSSLKFVPKGKEAAASQGH

SECIS Design:

Position of Selenocysteine:* 95 Help/Example

Amino Acid Conditions: Help/Example (conserved pos.)

SECIS-Element: Help FdhF-std(optional)

Define your own RNA-Element:

Structure: Help/Example

Nucleotides: Help/Example

Similarity BLOSUM62

Insertion Penalty Help -10

Deletion Penalty Help -5

RNAinverse (local search):

Search Strategy: Help Full Local Search

Objective Function: Help mfe -> nc

Valid Similarity Fraction: Help 0.7

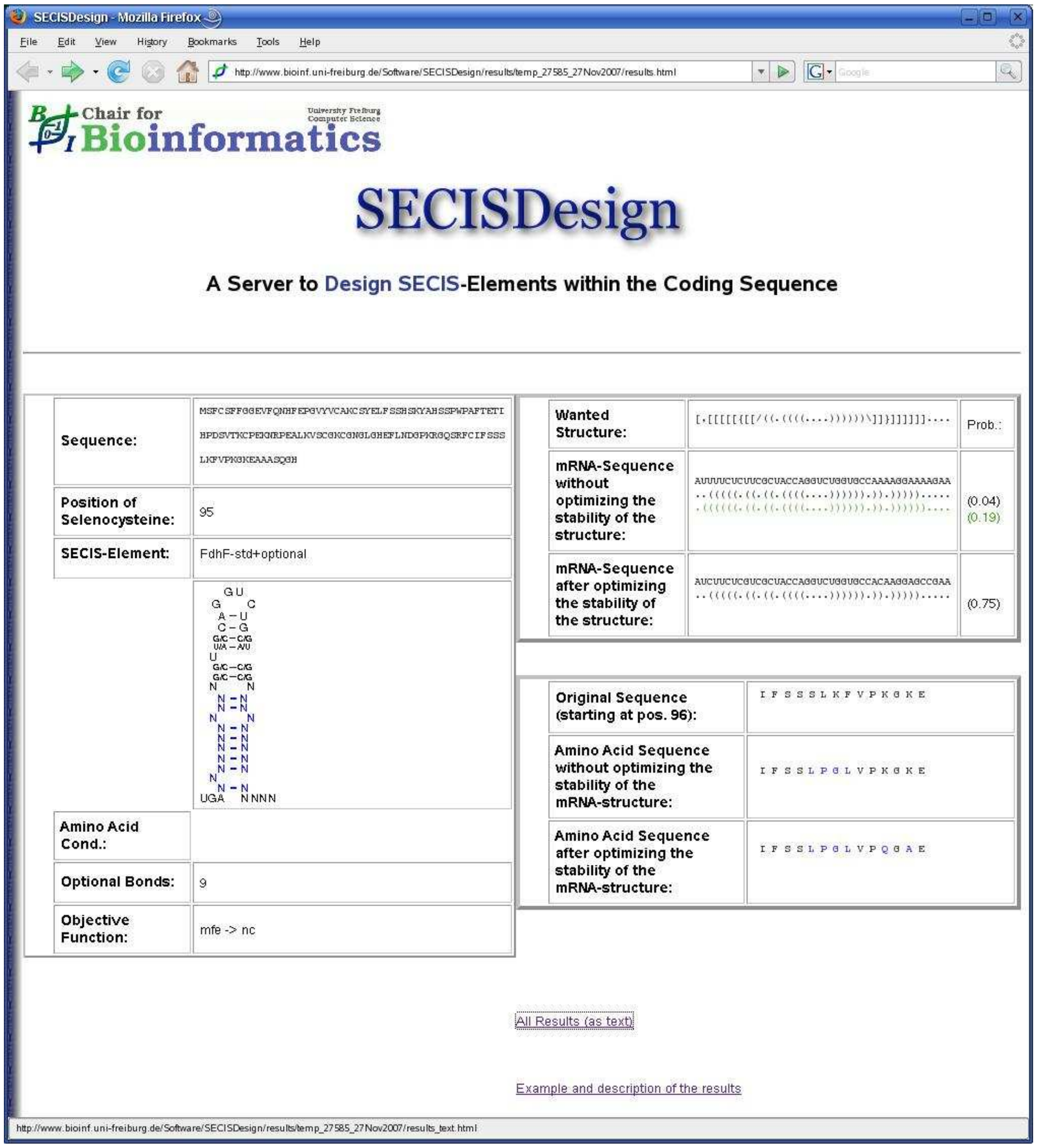
Compared Similarity: Help Start Similarity

Probabilities of Bases: Help all equal

Email address:* abusch@informatik.uni-freiburg.de

reset all submit all

Figure 4.3: The figure shows the input page of the SECISDesign web server, when inserting the SECIS-element shown in Figure 3.11A into the mRNA coding for MsrB in mouse in order to insert a selenocysteine at position 95.



http://www.bioinf.uni-freiburg.de/Software/SECISDesign/results/temp_27585_27Nov2007/results.html

SECISDesign

A Server to Design SECIS-Elements within the Coding Sequence

Sequence:	MSFCSFFGGEVFQNHFEPGVYVCAKCSYELFSSHSKYAHSSPWPAFTETI HPDSVTKCPEKNRPEALKVSCGKCGNGLGHEFLNDGPKRGQSRFCIFSSS LKFVPKGKEAAASQGH
Position of Selenocysteine:	95
SECIS-Element:	FdhF-std+optional
	see structure below
Amino Acid Cond.:	
Optional Bonds:	9
Objective Function:	mfe -> nc

```
    G U
  G     C
   A – U
   C – G
  G/C – C/G
  U/A – A/U
 U
  G/C – C/G
  G/C – C/G
 N       N
   N - N
   N - N
 N       N
   N - N
   N - N
   N - N
   N - N
   N - N
 N
   N - N
UGA    N NNN
```

Wanted Structure:	[.[[[[[{[[/((.((((....))))))\]]}]]]]]]....	Prob.:
mRNA-Sequence without optimizing the stability of the structure:	AUUUUCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAAAAGAA ..(((((.((.((.((((....)))))).)).)))))..... .((((((.((.((.((((....)))))).)).))))))....	(0.04) (0.19)
mRNA-Sequence after optimizing the stability of the structure:	AUCUUCUCGUCGCUACCAGGUCUGGUGCCACAAGGAGCCGAA ..(((((.((.((.((((....)))))).)).))))).....	(0.75)

Original Sequence (starting at pos. 96):	I F S S S L K F V P K G K E
Amino Acid Sequence without optimizing the stability of the mRNA-structure:	I F S S L P G L V P K G K E
Amino Acid Sequence after optimizing the stability of the mRNA-structure:	I F S S L P G L V P Q G A E

All Results (as text)

Example and description of the results

http://www.bioinf.uni-freiburg.de/Software/SECISDesign/results/temp_27585_27Nov2007/results_text.html

Figure 4.4: The figure shows the SECISDesign web server output of the computation requested in Figure 4.3.

Chapter 5

Conclusion

In this thesis, we introduced two approaches dealing with RNA sequence design. In the first part, we developed a fast and successful new approach to the inverse RNA folding problem, called INFO-RNA, while in the second part, our interests focused on a more complex design of RNA sequences overlapping the coding part. Thus, we have to care about the translated amino acid sequence additionally. We developed an algorithm, called SECISDesign, that solves this complex task successfully.
Both algorithms consist of two major steps: a new method of finding a good initialization via dynamic programming and a subsequent advanced local search. For both algorithms, we showed that the initialization provides a good starting sequence for the subsequent local search.

The approach discussed in the first part, INFO-RNA, finds an RNA sequence S that folds into a given secondary structure T and fulfills some given constraints C on the primary sequence. During its initialization step, INFO-RNA finds a sequence that among all valid sequences adopts the target structure with the lowest possible energy. There is no other sequence that has lower energy when folding into this structure. Nevertheless, the sequence is not guaranteed to fold into T since actually this sequence can have an even lower energy when folding into another structure. Therefore, the resulting sequence is processed further in a second step. During this second step of INFO-RNA, an advanced stochastic local search, which uses an effective neighbor selection method, improves the designed sequence concerning its foldability.

INFO-RNA runs fast, since the initializing sequence is found in linear time depending on the structure size and only few local search steps are needed to generate a good sequence that folds into the target structure.

In order to test the performance of INFO-RNA, artifically generated as well as biological RNA structures were analyzed. The results obtained when using INFO-RNA were compared to the results of the two existing tools RNA-SSD and RNAinverse. Generally, INFO-RNA outperforms RNA-SSD and performs substantially better than RNAinverse.
However in most cases, the initialization sequences of INFO-RNA have a high GC content. This is due to the fact that G-C base pairs are energetically more favorable compared to A-U or G-U pairs. Although the GC content is reduced during the local search, the final sequences are still enriched in G's and C's. This fact might also explain their high stability. In the future, it is desirable to offer special constraints that limit the GC content of sequences designed by INFO-RNA. Furthermore, it might be useful to design sequences whose stability is comparable to the stability of biological sequences folding into this structure since not only stability but also flexibility might be of interest.

The approach described in the second part, SECISDesign, deals with a more complex design problem: the design of sequences that fold into a given structure and code for a given protein. Although we used our approach to design a SECIS-element at a position in the coding part of an mRNA where a selenocysteine should be inserted, it can also be applied to the design of other RNA secondary structure elements located in the coding sequence of an mRNA.
We have given an RNA sequence constraints vector $C = C_1 ... C_{3n}$, an RNA secondary structure T, and an amino acid sequence $A = A_1 ... A_n$ of the protein where the selenocysteine is to be inserted at position A_0. SECISDesign also consists of two steps. During the first step, an RNA sequence $S = S_1 ... S_{3n}$ is designed that is optimal with respect to the combination of

(i) its similarity to the SECIS sequence constraints C and

(ii) the similarity its encoded amino acid sequence $A' = A'_1 ... A'_n$ has compared to the original amino acid sequence A of the protein.

Furthermore, the typical hairpin-stem structure T of a SECIS-element can be adopted. Since, during the first step, the folding properties of the SECIS-element are not taken into account, a second step is needed to mutate the sequence further in order to enhance its folding propability, but nevertheless, ensure a minimal similarity. Here, a local search in performed. The user can choose among several objective function concerning the foldability as well as among several local search methods, e.g. stochastic local search and full local search.

To test the performance of SECISDesign, we addressed the following two applications:

(a) in order to express a mammalian selenoprotein in *E.coli* and

(b) to insert selenocysteine in a protein, that has no selenocysteine at the considered position naturally, but a simple cysteine.

In both cases, we designed a typical SECIS-element of *E.coli* (*FdhF*) directly after the UGA codon with only a few changes within the amino acid sequence to allow its expression in *E.coli*. For most of the proteins, we could design stable SECIS-elements and nevertheless preserve the proteins with only few changes in their amino acid sequences.

While the full local search is successfully used in SECISDesign, INFO-RNA does not benefit from it. We performed several tests of using the full local search as well as the adaptive walk also during INFO-RNA, but we obtained bad and slow results (data not shown). Here, the stochastic local search provides the best results. This might be due to the large amounts of tested sequences when using the full local search or to getting stuck in a local optimum when using the adaptive walk.

To conclude, both, INFO-RNA and SECISDesign, are fast and successful tools for the design of RNA sequences dealing with simple and complex constraints, respectively. They outperform existing tools and methods for most structures.

Bibliography

[ABK$^+$97] K. J. Addess, J. P. Basilion, R. D. Klausner, T. A. Rouault, and A. Pardi. Structure and dynamics of the iron responsive element RNA: implications for binding of the RNA by iron regulatory binding proteins. *Journal of Molecular Biology*, 274(1):72–83, 1997.

[AFH$^+$04] M. Andronescu, A.P. Fejes, F. Hutter, H.H. Hoos, and A. Condon. A new algorithm for RNA secondary structure design. *Journal of Molecular Biology*, 336(3):607–624, 2004.

[AHHC07] R. Aguirre-Hernandez, H. H. Hoos, and A. Condon. Computational RNA secondary structure design: empirical complexity and improved methods. *BMC Bioinformatics*, 8:34, 2007.

[AMK$^+$00] M. Antal, A. Mougin, M. Kis, E. Boros, G. Steger, G. Jakab, F. Solymosy, and C. Branlant. Molecular characterization at the RNA and gene levels of U3 snoRNA from a unicellular green alga, Chlamydomonas reinhardtii. *Nucleic Acids Research*, 28(15):2959–68, 2000.

[ASL$^+$99] E. S. Arner, H. Sarioglu, F. Lottspeich, A. Holmgren, and A. Böck. High-level expression in Escherichia coli of selenocysteine-containing rat thioredoxin reductase utilizing gene fusions with engineered bacterial-type SECIS elements and co-expression with the selA, selB and selC genes. *Journal of Molecular Biology*, 292(5):1003–16, 1999.

[BA01] K. M. Brown and J. R. Arthur. Selenium, selenoproteins and human health: a review. *Public Health Nutr*, 4(2B):593–9, 2001.

[BB04] R. Backofen and A. Busch. Computational design of new and recombinant selenoproteins. In *Proc. of the 15th Annual Symposium on Combinatorial Pattern Matching (CPM2004)*, pages 270–284, 2004.

[BB06] A. Busch and R. Backofen. INFO-RNA–a fast approach to inverse RNA folding. *Bioinformatics*, 22(15):1823–31, 2006.

[BB07] A. Busch and R. Backofen. INFO-RNA–a server for fast inverse RNA folding satisfying sequence constraints. *Nucleic Acids Research*, 35(Webserver issue):W310–313, 2007.

[BFHB91] A. Böck, K. Forchhammer, J. Heider, and C. Baron. Selenoprotein synthesis: an expansion of the genetic code. *Trends Biochem Sci*, 16(12):463–467, 1991.

[BFHV05] G. Blin, G. Fertin, D. Hermelin, and S. Vialette. Fixed-parameter algorithms for protein search under mRNA structure constraints. In *Proc. of Graph-Theoretical Concepts in Computer Science*, volume 3787 of *LNCS*, pages 271–282. Springer, 2005.

[BHB93] C. Baron, J. Heider, and A. Böck. Interaction of translation factor SELB with the formate dehydrogenase H selenopolypeptide mRNA. *Proc. Natl. Acad. Sci. USA*, 90(9):4181–5, 1993.

[BHS06] S. H. Bernhart, I. L. Hofacker, and P. F. Stadler. Local RNA base pairing probabilities in large sequences. *Bioinformatics*, 22(5):614–5, 2006.

[BJ90] A. A. Beaudry and G. F. Joyce. Minimum secondary structure requirements for catalytic activity of a self-splicing group I intron. *Biochemistry*, 29(27):6534–9, 1990.

[BKML$^+$07] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler. GenBank. *Nucleic Acids Research*, 35(Database issue):D21–5, 2007.

[BNM02] S. Bar-Noy and J. Moskovitz. Mouse methionine sulfoxide reductase B: effect of selenocysteine incorporation on its activity and expression of the seleno-containing enzyme in bacterial and mammalian cells. *Biochem Biophys Res Commun*, 297(4):956–61, 2002.

[BNS02a] R. Backofen, N. S. Narayanaswamy, and F. Swidan. Protein similarity search under mRNA structural constraints: application to targeted selenocysteine insertion. *In Silico Biology*, 2(3):275–90, 2002.

[BNS02b] R. Backofen, N.S. Narayanaswamy, and F. Swidan. On the complexity of protein similarity search under mRNA structure constraints. In *Proc. of 19th International Symposium on Theoretical Aspects of Computer Science (STACS2002)*, volume 2285 of *LNCS*, pages 274–286, Berlin, 2002. Springer.

[Bon04] D. Bongartz. Some notes on the complexity of protein similarity search under mRNA structure constraints. In *Proc. of the 30th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 2932 of *LNCS*, pages 174–183. Springer, 2004.

[BWB05] A. Busch, S. Will, and R. Backofen. SECISDesign: a server to design SECIS-elements within the coding sequence. *Bioinformatics*, 21(15):3312–3, 2005.

[CB85] A. Cornish-Bowden. Nomenclature for incompletely specified bases in nucleic acid sequences: recommendations 1984. *Nucleic Acids Research*, 13(9):3021–30, 1985.

[CCF$^+$05] J. R. Cole, B. Chai, R. J. Farris, Q. Wang, S. A. Kulam, D. M. McGarrell, G. M. Garrity, and J. M. Tiedje. The Ribosomal Database Project (RDP-II): sequences and tools for high-throughput rRNA analysis. *Nucleic Acids Research*, 33(Database issue):D294–6, 2005.

[CCM$^+$03] J. R. Cole, B. Chai, T. L. Marsh, R. J. Farris, Q. Wang, S. A. Kulam, S. Chandra, D. M. McGarrell, T. M. Schmidt, G. M. Garrity, and J. M. Tiedje. The Ribosomal Database Project (RDP-II): previewing a new autoaligner that allows regular updates and the new prokaryotic taxonomy. *Nucleic Acids Research*, 31(1):442–3, 2003.

[Cec92] T. R. Cech. Ribozyme engineering. *Current Opinion in Structural Biology*, 2(4):605–9, 1992.

[Cec04] T. R. Cech. RNA finds a simpler way. *Nature*, 428(6980):263–4, 2004.

[CO00] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101:77–114, 2000.

[Cri68] F. H. Crick. The origin of the genetic code. *Journal of Molecular Biology*, 38(3):367–79, 1968.

[DE04] R. D. Dowell and S. R. Eddy. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics*, 5(1):71, 2004.

[DF99] R. Downey and M. Fellows. *Parameterized Complexity.* Springer, 1999.

[DLWP04] R. M. Dirks, M. Lin, E. Winfree, and N. A. Pierce. Paradigms for computational nucleic acid design. *Nucleic Acids Research*, 32(4):1392–403, 2004.

[DS89] J. A. Doudna and J. W. Szostak. Miniribozymes, small derivatives of the sunY intron, are catalytically active. *Mol Cell Biol*, 9(12):5480–3, 1989.

[DWB06] C. B. Do, D. A. Woods, and S. Batzoglou. CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics*, 22(14):e90–8, 2006.

[Fed00] M. J. Fedor. Structure and function of the hairpin ribozyme. *Journal of Molecular Biology*, 297(2):269–91, 2000.

[FFHS00] C. Flamm, W. Fontana, I. L. Hofacker, and P. Schuster. RNA folding at elementary step resolution. *RNA*, 6(3):325–38, 2000.

[FKJ$^+$86] S. M. Freier, R. Kierzek, J. A. Jaeger, N. Sugimoto, M. H. Caruthers, T. Neilson, and D. H. Turner. Improved free-energy parameters for predictions of RNA duplex stability. *Proc. Natl. Acad. Sci. USA*, 83(24):9373–7, 1986.

[FLB89] K. Forchhammer, W. Leinfelder, and A. Böck. Identification of a novel translation factor necessary for the incorporation of selenocysteine into protein. *Nature*, 342(6248):453–6, 1989.

[GG04] P. P. Gardner and R. Giegerich. A comprehensive comparison of comparative RNA structure prediction approaches. *BMC Bioinformatics*, 5:140, 2004.

[Gil86] W. Gilbert. The RNA world. *Nature*, 319:618, 1986.

[GTGM$^+$83] C. Guerrier-Takada, K. Gardiner, T. Marsh, N. Pace, and S. Altman. The RNA moiety of ribonuclease P is the catalytic subunit of the enzyme. *Cell*, 35(3 Pt 2):849–57, 1983.

[Gur07] F. Gurski. Polynomial algorithms for protein similarity search for restricted mRNA structures. *Computing Research Repository*, http://arxiv.org/abs/0704.3496 Apr 2007.

[HAV$^+$96] E. S. Haas, D. W. Armbruster, B. M. Vucson, C. J. Daniels, and J. W. Brown. Comparative analysis of ribonuclease P RNA structure in Archaea. *Nucleic Acids Research*, 24(7):1252–9, 1996.

[HB98] A. Hüttenhofer and A. Böck. RNA structures involved in selenoprotein synthesis. In *RNA Structure and Function*, pages 603–639. Cold Spring Habor Laboratory Press, Cold Spring Habor, 1998.

[HBB02] A. Hüttenhofer, J. Brosius, and J. P. Bachellerie. RNomics: identification and function of small, non-messenger RNAs. *Current Opinion in Chemical Biology*, 6(6):835–43, 2002.

[HBS04] I. L. Hofacker, S. H. Bernhart, and P. F. Stadler. Alignment of RNA base pairing probability matrices. *Bioinformatics*, 2004.

[HCF$^+$00] S. Hazebrouck, L. Camoin, Z. Faltin, A. D. Strosberg, and Y. Eshdat. Substituting selenocysteine for catalytic cysteine 41 enhances enzymatic activity of plant phospholipid hydroperoxide glutathione peroxidase expressed in Escherichia coli. *Journal of Biological Chemistry*, 275(37):28715–28721, 2000.

[HFS$^+$94] I. L. Hofacker, W. Fontana, P. F. Stadler, S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures. *Monatshefte Chemie*, 125:167–188, 1994.

[HK96] M. W. Hentze and L. C. Kuhn. Molecular control of vertebrate iron metabolism: mRNA-based regulatory circuits operated by iron, nitric oxide, and oxidative stress. *Proc. Natl. Acad. Sci. USA*, 93(16):8175–82, 1996.

[Hof94] I. L. Hofacker. *The Rules of the Evolutionary Game for RNA: A Statistical Characterization of the Sequence to Structure Mapping in RNA*. PhD thesis, Vienna University, 1994.

[Hoo98] H. H. Hoos. *Stochastic Local Search - Methods, Models, Applications.* PhD thesis, Darmstadt University of Technology, 1998.

[HWB96] A. Hüttenhofer, E. Westhof, and A. Böck. Solution structure of mRNA hairpins promoting selenocysteine incorporation in Escherichia coli and their base-specific interaction with special elongation factor SELB. *RNA*, 2(4):354–66, 1996.

[JDR00] L. Jovine, S. Djordjevic, and D. Rhodes. The crystal structure of yeast phenylalanine tRNA at 2.0 A resolution: cleavage by Mg(2+) in 15-year old crystals. *Journal of Molecular Biology*, 301(2):401–14, 2000.

[JTZ89] J. A. Jaeger, D. H. Turner, and M. Zuker. Improved predictions of secondary structures for RNA. *Proc. Natl. Acad. Sci. USA*, 86(20):7706–10, 1989.

[KAA+07] T. Kulikova, R. Akhtar, P. Aldebert, N. Althorpe, M. Andersson, A. Baldwin, K. Bates, S. Bhattacharyya, L. Bower, P. Browne, M. Castro, G. Cochrane, K. Duggan, R. Eberhardt, N. Faruque, G. Hoad, C. Kanz, C. Lee, R. Leinonen, Q. Lin, V. Lombard, R. Lopez, D. Lorenc, H. McWilliam, G. Mukherjee, F. Nardone, M. P. G. Pastor, S. Plaister, S. Sobhany, P. Stoehr, R. Vaughan, D. Wu, W. Zhu, and R. Apweiler. EMBL Nucleotide Sequence Database in 2006. *Nucleic Acids Research*, 35(Database issue):D16–20, 2007.

[KCN+03] 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.

[KGZ+82] K. Kruger, P. J. Grabowski, A. J. Zaug, J. Sands, D. E. Gottschling, and T. R. Cech. Self-splicing RNA: autoexcision and autocyclization of the ribosomal RNA intervening sequence of Tetrahymena. *Cell*, 31(1):147–57, 1982.

[KH99] B. Knudsen and J. Hein. RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics*, 15(6):446–54, 1999.

[KH03] B. Knudsen and J. Hein. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Research*, 31(13):3423–8, 2003.

[Kni03] J. Knight. Gene regulation: switched on to RNA. *Nature*, 425(6955):232–3, 2003.

[LB96] 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.

[LNL01] D. A. Lafontaine, D. G. Norman, and D. M. Lilley. Structure, folding and activity of the VS ribozyme: importance of the 2-3-6 helical junction. *EMBO J*, 20(6):1415–24, 2001.

[LRGEK98] Z. Liu, M. Reches, I. Groisman, and H. Engelberg-Kulka. The nature of the minimal 'selenocysteine insertion sequence' (SECIS) in Escherichia coli. *Nucleic Acids Research*, 26(4):896–902, 1998.

[LSW02] N. B. Leontis, J. Stombaugh, and E. Westhof. Motif prediction in ribosomal RNAs Lessons and prospects for automated motif prediction in homologous RNA molecules. *Biochimie*, 84(9):961–73, 2002.

[LTM06] Z. J. Lu, D. H. Turner, and D. H. Mathews. A set of nearest neighbor parameters for predicting the enthalpy change of RNA secondary structure formation. *Nucleic Acids Research*, 34(17):4912–24, 2006.

[LZMBB88] W. Leinfelder, E. Zehelein, M. A. Mandrand-Berthelot, and A. Böck. Gene for a novel tRNA species that accepts L-serine and cotranslationally inserts selenocysteine. *Nature*, 331(6158):723–5, 1988.

[LZP99] R. B. Lyngso, M. Zuker, and C. N. Pedersen. Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics*, 15(6):440–5, 1999.

[Mac92] G. A. Mackie. Secondary structure of the mRNA for ribosomal protein S20. Implications for cleavage by ribonuclease E. *Journal of Biological Chemistry*, 267(2):1054–61, 1992.

[Mar84] H. M. Martinez. An RNA folding rule. *Nucleic Acids Research*, 12(1 Pt 1):323–34, 1984.

[McC90] J. S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–19, 1990.

[MDK85] A. A. Mironov, L. P. Dyakonova, and A. E. Kister. A kinetic approach to the prediction of RNA secondary structures. *J Biomol Struct Dyn*, 2(5):953–62, 1985.

[MM06] J. S. Mattick and I. V. Makunin. Non-coding RNA. *Hum Mol Genet*, 15 Spec No 1:R17–29, 2006.

[MP99] E. M. Mobley and T. Pan. Design and isolation of ribozyme-substrate pairs using RNase P-based ribozymes containing altered substrate binding sites. *Nucleic Acids Research*, 27(21):4298–304, 1999.

[MSZT99] D.H. Mathews, J. Sabina, M. Zuker, and D.H. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J Mol Biol*, 288(5):911–40, 1999.

[MZS$^+$00] V. Moulton, M. Zuker, M. Steel, R. Pointon, and D. Penny. Metrics on RNA secondary structures. *Journal of Computational Biology*, 7(1-2):277–292, 2000.

[NHZ92] H. F. Noller, V. Hoffarth, and L. Zimniak. Unusual resistance of peptidyl transferase to protein extraction procedures. *Science*, 256(5062):1416–9, 1992.

[OC93] L. E. Orgel and F. H. Crick. Anticipating an RNA world. Some past speculations on the origin of life: where are they today? *FASEB J*, 7(1):238–9, 1993.

[Org68] L. E. Orgel. Evolution of the genetic apparatus. *Journal of Molecular Biology*, 38(3):381–93, 1968.

[PGPZP98] C. Pitulle, M. Garcia-Paris, K. R. Zamudio, and N. R. Pace. Comparative structure analysis of vertebrate ribonuclease P RNA. *Nucleic Acids Research*, 26(14):3333–9, 1998.

[San85] D. Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J. Appl. Math.*, 45(5):810–825, 1985.

[SBEB02] M. Szymanski, M. Z. Barciszewska, V. A. Erdmann, and J. Barciszewski. 5s Ribosomal RNA Database. *Nucleic Acids Research*, 30(1):176–8, 2002.

[SCGZ02] L. Sun, Z. Cui, R. L. Gottlieb, and B. Zhang. A selected ribozyme catalyzing diverse dipeptide synthesis. *Chem Biol*, 9(5):619–28, 2002.

[SDSP01] J. Swisher, C. M. Duarte, L. J. Su, and A. M. Pyle. Visualizing the solvent-inaccessible core of a group II intron ribozyme. *EMBO J*, 20(8):2051–61, 2001.

[SEB07] M. Szymanski, V. A. Erdmann, and J. Barciszewski. Noncoding RNAs database (ncRNAdb). *Nucleic Acids Research*, 35(Database issue):D162–4, 2007.

[SFSH94] P. Schuster, W. Fontana, P. F. Stadler, and I. L. Hofacker. From sequences to shapes and back: a case study in RNA secondary structures. *Proc. Royal Society London B*, 255(1344):279–84, 1994.

[SHZB91] G. Sawers, J. Heider, E. Zehelein, and A. Böck. Expression and operon structure of the sel genes of escherichia coli and identification of a third selenium-containing formate dehydrogenase isoenzyme. *J Bacteriol.*, 173(16):4983–93, 1991.

[SOR$^+$06] R. Schwab, S. Ossowski, M. Riester, N. Warthmann, and D. Weigel. Highly specific gene silencing by artificial microRNAs in Arabidopsis. *Plant Cell*, 18(5):1121–33, 2006.

[SP07] A. Serganov and D. J. Patel. Ribozymes, riboswitches and beyond: regulation of gene expression without proteins. *Nat Rev Genet*, 8(10):776–90, 2007.

[Sto02] G. Storz. An expanding universe of noncoding RNAs. *Science*, 296(5571):1260–3, 2002.

[THB00] R. M. Tujebajeva, J. W. Harney, and M. J. Berry. Selenoprotein P expression, purification, and immunochemical characterization. *Journal of Biological Chemistry*, 275(9):6288–94, 2000.

[TS88] D. H. Turner and N. Sugimoto. RNA structure prediction. *Annu Rev Biophys Biophys Chem*, 17:167–92, 1988.

[TSJ$^+$87] D. H. Turner, N. Sugimoto, J. A. Jaeger, C. E. Longfellow, S. M. Freier, and R. Kierzek. Improved parameters for prediction of RNA structure. *Cold Spring Harb Symp Quant Biol*, 52:123–33, 1987.

[UM95] N. Usman and J.A. McSwiggen. Catalytic RNA (ribozymes) as drugs. In *Annual Reports on Medicinal Chemistry*, volume 30, pages 285–294. Academic Press, 1995.

[VGM$^+$00] L. Varani, S. I. Gunderson, I. W. Mattaj, L. E. Kay, D. Neuhaus, and G. Varani. The NMR structure of the 38 kDa U1A protein - PIE RNA complex reveals the basis of cooperativity in regulation of polyadenylation by human U1A protein. *Nat Struct Biol*, 7(4):329–35, 2000.

[Wat78] M. S. Waterman. Secondary structure of single-stranded nucleic acids. *Studies on foundations and combinatorics, Advances in mathematics supplementary studies*, 1:167 – 212, 1978.

[WC53] J. D. Watson and F. H. Crick. Molecular structure of nucleic acids. a structure for deoxyribose nucleic acid. *Nature*, 171:737–741, 1953.

[WMJ96] E. Westhof, B. Masquida, and L. Jaeger. RNA tectonics: towards RNA design. *Fold Des*, 1(4):R78–88, 1996.

[WNR+04] W. C. Winkler, A. Nahvi, A. Roth, J. A. Collins, and R. R. Breaker. Control of gene expression by a natural metabolite-responsive ribozyme. *Nature*, 428(6980):281–6, 2004.

[Woo00] S. A. Woodson. Recent insights on RNA folding mechanisms from catalytic RNA. *Cell Mol Life Sci*, 57(5):796–808, 2000.

[WRH+07] S. Will, K. Reiche, I. L. Hofacker, P. F. Stadler, and R. Backofen. Inferring non-coding RNA families and classes by means of genome-scale structure-based clustering. *PLOS Computational Biology*, 3(4):e65, 2007.

[WS95] C. Wilson and J. W. Szostak. In vitro evolution of a self-alkylating ribozyme. *Nature*, 374(6525):777–82, 1995.

[ZHB90] F. Zinoni, J. Heider, and A. Böck. Features of the formate dehydrogenase mRNA necessary for decoding of the UGA codon as selenocysteine. *Proc. Natl. Acad. Sci. USA*, 87(12):4660–4, 1990.

[ZJT91] M. Zuker, J. A. Jaeger, and D. H. Turner. A comparison of optimal and suboptimal RNA secondary structures predicted by free energy minimization with structures determined by phylogenetic comparison. *Nucleic Acids Research*, 19(10):2707–14, 1991.

[ZMT99] M. Zuker, D. H. Mathews, and D. H. Turner. Algorithms and thermodynamics for RNA secondary structure prediction: A practical guide. In *RNA Biochemistry and Biotechnology*, pages 11–43. Kluwer Academic Publishers, Dordrecht, NL, 1999.

[ZS81] M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research*, 9(1):133–48, 1981.

[Zuk94] M. Zuker. Prediction of RNA secondary structure by energy minimization. *Methods in Molecular Biology*, 25:267–94, 1994.

[Zuk03] M. Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Research*, 31(13):3406–15, 2003.

Abbreviations

A	adenine
AA	amino acid
AW	adaptive walk
BL	bulge loop
C	cytosine
CPU	central processing unit
DP	dynamic programming
EL	exterior loop
FLS	full local search
G	guanine
HL	hairpin loop
IL	interior loop
IRE	iron responsive element
mfe	minimum free energy
miRNA	micro RNA
ML	multiloop
mRNA	messenger RNA
ncRNA	non-coding RNA
PIE	polyadenylation inhibition element
rRNA	ribosomal RNA
SECIS	selenocysteine insertion sequence
siRNA	small interfering RNA
SLS	stochastic local search
snoRNA	small nucleolar RNA
snRNA	small nuclear RNA
tRNA	transfer RNA
U	uracil
UTR	untranslated region
w.r.t.	with respect to

Index