Université de Montréal

Recherche de structures secondaires dans les séquences biologiques

Par

Houjing Huang

Département d'informatique et de recherche opérationelle
Faculté des Arts et Sciences

Mémoire présenté à la faculté des études supérieures
en vue de l'obtention du grade de
Maître ès science (M.Sc.)
en Informatique

Novembre, 2001

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé:

Recherche de structures secondaires
dans les séquences biologiques

Présenté par:

Houjing Huang

A été évalué par un jury composé des personnes suivantes:

Bernard Gendron,	Président-rapporteur
Nadia El-Mabrouk,	Directeure de recherche
Miklós Csürös,	Membre du jury

Mémoire accepté le 19-12-2001

Résumé

L'un des défis de la biologie moléculaire est d'identifier les gènes et les autres éléments génétiques dans des séquences d'ADN d'une longueur de plusieurs milliers et même de plusieurs millions de nucléotides. Du fait de la grande quantité d'ADN séquencée chaque année, le décodage de l'information génétique ne peut pas reposer sur la seule utilisation de méthodes expérimentales, et l'utilisation d'algorithmes de recherche devient indispensable. Plusieurs méthodes ont été développées pour la recherche de motifs biologiques plus ou moins complexes. L'inconvénient majeur de ces méthodes est le manque de flexibilité dans la définition des motifs à rechercher.

Quelle que soit la méthode utilisée, elle est toujours basée sur la recherche de sous-structures primaires ou secondaires conservées. Les sous-structures secondaires conservées sont généralement des hélices formées d'un bras d'appariements et d'une boucle. L'algorithme de Sagot-Viari est l'un des algorithmes les plus appropriés pour la recherche de telles hélices. Son avantage majeur est de permettre toutes sortes d'incertitudes dans le motif (insertions, suppressions, substitution de nucléotides ou de paires de bases). Les hélices sont définies par les tailles maximales et minimales de boucle et de bras, et le nombre maximal d'erreurs autorisées. Dans ce mémoire, nous généralisons cet algorithme à toutes sortes d'hélices comprenant des boucles internes et panse, des bases invariantes et semi-invariantes, et nous utilisons des scores pour filtrer les résultats. Nous appliquons l'algorithme à la recherche des gènes d'ARN 5S et de RNase P RNA dans différents génomes.

Mots klef: hélice, ARN, structure secondaire, génome, algorithme de recherche.

Abstract

One of the challenges of molecular biology is to identify the different genes and other genetic elements that play a major role in the biological function of the organism. Due to the large variety of genomes sequenced each year, DNA analysis cannot rely exclusively on experimental methods, and a preliminary computer processing of sequences is usually considered. Several methods have been developed for identifying more or less complex RNA structures in a genome. Each of these methods has its advantages and drawbacks, but the general problem is the lack of flexibility in defining the structure to be searched for.

Whatever the method is, it is always based on the identification of various conserved primary and secondary sub-structures called helices. One of the most appropriate algorithms for identifying helices (stem-loop structures) is the Sagot-Viari algorithm, as it allows for various kinds of uncertainties (insertion, deletion and substitution of nucleotides and base-pairs). The helices in the Sagot-Viari algorithm are defined by a loop length, a step length, and a maximal number of allowed errors. In this dissertation, we extend the Sagot-Viari algorithm to allow for searching different kinds of helices with possible preserved internal loops and bulges, conserved or semi-conserved nucleotides, and various kinds of scores. We use our new algorithm to identify the 5S RNA genes and the RNase P RNAs in different genomes.

Keywords: helix, RNA, secondary structure, genomic sequence, searching algorithm.

Table of contents

List of Figures

List of Tables

Acknowledgments

First of all, I wish to express my gratitude towards my director, Dr. Nadia El-Mabrouk, for her accurate advice and for her patience. I am also grateful for the great latitude and confidence that she granted to me in the direction of my work.

I would like to stress the valuable help from Dr. Marie-France Sagot, for her early work on this aspect and for the useful discussions.

I also would like to thank Dr. Gertraud Burger and Dr. B. Franz Lang for their help and for giving me the opportunity to test my algorithm on their database and bacterial genome sequences.

Chapter 1

Introduction

The genetic inheritance of any organism is contained in its genome made up of one or several DNA molecules. Each molecule can be seen as a sequence of four nucleotides: A, C, G, T. This observation that the DNA contains all the "instructions" responsible of the biological functions of an organism as well as the mechanism to carry out these instructions is known as the "Central Dogma of molecular biology". The genes are the parts of the DNA molecule that contain all the genetic information. In particular, "coding sequences" are the genes containing the instructions for protein synthesis. These macromolecules play a major role in the genetic machinery.

The general mechanism for decoding the "genetic message" follows two major steps. The first one is the "transcription" of the DNA into RNA. More precisely, the coding regions of the genome are "copied" into RNA molecules (mRNAs, tRNAs and ribosomal RNAs). The second step is the "translation". The genetic message contained in the mRNA is then "read" and "translated" into proteins. The primary structure of the RNA is a sequence of four nucleotides: A, C, G, U. The molecule folds into a secondary and tertiary structure, by forming hydrogen bonds between nucleotides A, U, and nucleotides G, C. The paired regions of the RNA are called helices.

The challenge of molecular biology is to understand the genetic message contained in the DNA. In particular, the goal is to identify the different genes and other genetic elements that play a major role in the biological function of the organism. Such coding sequences can be of several thousands and even of several million nucleotides long. Because of the huge amount of information contained in the databases and the large variety of new genomes sequenced each year, DNA analysis cannot rely exclusively on experimental methods, and a preliminary computer processing of sequences is usually considered. For example, whenever a new gene is sequenced, the first task is to use a research tool such as FASTA [PL88] or BLAST [AGM90] to search for homologous genes in the existing genomic databases. More sophisticated algorithms for searching more complex RNA structures have also been developed. Such algorithms are used to "filter" the DNA parts that can possibly contain specific genes. Experimental methods can then be used to verify such hypotheses.

According to different empirical methods, biochemical techniques, multiple sequence alignment and dynamic programming algorithms, some common secondary structures of certain RNA families and other genetic elements have been determined. In particular, tRNA molecules have been extensively studied. The corresponding gene sequence is short (about 75 nucleotides), and the primary, secondary and even tertiary structure is very conserved [RRB76]. Other more complex and less constrained structures have been studied, such as group I and II introns [LDM94, KKB94, MUO89], bacterial RNase P RNA [MJW98, LBK97] and various ribosomal RNAs. However, in most cases, the consensus structure is very difficult to establish, and only some characteristics of the primary and secondary structure are known.

Several methods have been developed for identifying more or less complex RNA structures. Some of them are "tailor-made" for searching specific gene families. For example, many algorithms have been developed to identify all tRNA genes in a

genome [FB91, EML96, LE97], and others have been developed for searching the Escherichia coli transcription terminator [dCBT90], snoRNAs [LE99, OLR00], and group I introns [LDM94]. The idea is to find the conditions that precisely and uniquely define a specific gene family, and to develop an algorithm that takes all these constraints into account. Other methods are more general and designed to search for any kind of constrained sequences described by the user in an input file. This is the case of RNAmot [GMC90], RNAbob[Edd96], Palingol[BKV96]. The major disadvantage of these methods is the lack of flexibility in defining the structures to be identified. For example, it is hard, if not impossible, to search for helices with potential internal loops and bulges (unpaired regions). More generally any difference between the most common structure and a variation of this structure is difficult to include in the definition of a consensus structure.

Whatever the method is, it is always based on the identification of various conserved primary and secondary sub-structures. Given an initial alignment of homologous sequences in different genomes, a conserved sub-structure is one approximately repeated at the same position in all the sequences of the alignment. As for conserved secondary sub-structures (stem-loop structures, pseudo-knots), they are defined by conserved nucleotides, stem length, loop length and possible bulges and internal loops. Our goal here is to develop a very flexible tool that will help the biologist in identifying all kinds of secondary sub-structures, with different kinds of deviations (errors).

One of the most flexible algorithms for identifying structural objects, such as helices (stem-loops), pseudoknots and triple helices is probably the Sagot-Viari algorithm [SV97]. The algorithm treats the problem of errors with the help of an object called a model against which the comparisons are made. In this case, a model is either a word or a pair of words over the same alphabet as that of the sequences that have both direct and inverse occurrences in the sequence. Moreover, errors (substitutions,

deletions and insertions) are allowed between a model and its inverse occurrences. Helix stems may, therefore, present bulges or interior loops. Reasonably efficient performance comes from the fact that the parts composing the structures are kept separated until the end and that filtering for valid occurrences (occurrences that may form part of such a structure) can be done in O(n) time where n is the length of the sequence. In the Sagot-Viari algorithm the helices are defined by a loop length, a stem length, and a maximal number of allowed errors.

In this dissertation, we extend the Sagot-Viari algorithm to search for different kinds of helices. Moreover, we improve the output of the algorithm by filtering the solutions and selecting the best results between all possible occurrences. These occurrences are chosen on the base of a general score. We first study the different helices that can be encountered in complex biological structures to better understand the needs, and to be able to develop a program that will be used in an efficient way to search for all kinds of structures. This preliminary study shows that a flexible definition of a helix should take into account potential internal loops and bulges, potential conserved nucleotides or subsets of nucleotides, as well as different distance measures on the primary and secondary structure (errors in the conserved nucleotides, or in the paired regions).

Such an algorithm for searching helices can be used in the different existing methods of complex biological structure search (RNAmot, RNAbob, Palingol) to improve the flexibility of these algorithms, or can be the basis of another general method. The idea is to first subdivide the general structure into a set of helices, to search for all these helices in the genome being analyzed by our algorithm, and then to assemble the different substructures. In that way, the user can try to assemble the "basic" helices in different ways, and new structures can be more easily discovered.

In Chapter 2, we present the basic concepts of molecular biology that we will use in the rest of this thesis. Chapter 3 is a review of various existing algorithms for identifying complex biological structures in a genome. In particular, we describe the Sagot-Viari algorithm for identifying helices in a genome. In Chapter 4, we first present different types of helices found in different types of RNA sequences and other genetic elements. In particular, we present conserved structures in tRNAs, mitochondrial 5S RNAs and mitochondrial RNase P RNAs. We then present our improvements to the Sagot-Viari algorithm. In Chapter 5, we present our applications and results for searching mitochondrial 5S RNA and mitochondrial RNase P RNA in various genomes. In each case, the algorithm identifies the annotated RNAs, with no false negatives and a few numbers of false positives.

Chapter 2

Basic concepts in molecular biology

In this chapter, we introduce the basic notions that are fundamental to understand the biological problem we are facing, and the concepts that we will use in the rest of this dissertation.

2.1 Biological background

The genetic information of an organism is stored in one or more distinct DNA molecules, called **chromosomes**. The set of all chromosomes of an organism is referred to as its **genome**. For example, the human genome contains 23 pairs of chromosomes. RNA molecules also store the genetic information of an organism. Some RNAs contribute to specific biochemical functions, whereas others, the messenger RNAs (mRNA), are the drivers of protein synthesis.

DNA (deoxyribonucleic acid) and RNA (ribonucleic acid) are long polymers of four types of monomers (small molecules) called **nucleotides**. Each nucleotide consists of three parts: one of two base molecules, plus a sugar, and one or more phosphate groups. The nucleotides differ by their base molecule. There are two types of bases: **purines** (denoted R) and **pyrimidines** (denoted Y). In the DNA, the purines are

Adenine (A) and **Guanine** (G), and the pyrimidines are **Cytosine** (C) and **Thymine** (T). In the DNA alphabet, U replaces T.

Nucleotides are sometimes called **bases**, and since DNA consists of two complementary strands bonded together, these units are often called **base-pairs**. A DNA molecule can be seen as a linear sequence on the alphabet of four letters N={A,C,G,T} (each letter corresponding to the nucleotide beginning by this letter), for example "AGATCAGG". A strand of DNA has a head (called the 5' end) and a tail (called the 3' end). The size of a DNA molecule is about 10^6 to 10^9 bases.

One well-known fact about DNA is that it forms a double helix, which is two helical (spiral-shaped) strands of the polypeptide, running in opposite directions, held together by hydrogen bonds. The nucleotide A bonds exclusively with T (or U in the case of the RNA) and forms the base-pair A-T, and the G bonds exclusively with C (G-C). A is the complementary base of T (and conversely), and C is the complementary base of G (and conversely).

In contrast with DNA, an RNA molecule is made up of only one strand. However, certain complementary parts of an RNA sequence are paired together to form a two-dimensional and three-dimensional structure. The paired regions are called **stems**. Although the most frequent base-pairs are Watson-Crick (A-U, C-G), other non-canonical pairings are also possible, the most frequent one being G-U. The function of an RNA is determined by its structure.

Proteins constitute a third category of macromolecules that play a major role in the genetic machinery. Proteins are responsible of most functions of a cell. They are enzymes and catalysts that drive the chemical reactions of the cell, they are the switches that control whether genes are turned on or off, they are the effectors that make muscles move. All proteins are made of the same basic constituents: the amino

acids. The amino acids are linked together by peptide bonds, and long chains of amino acids are strung together into polymers, called **polypeptides**. The length of a protein can vary from tens to thousands of amino acids. There are 20 different amino acids. Each one is encoded in the DNA by a sequence of three nucleotides, called a **codon**. Most amino acids are encoded by more than one codon. For example, alanine is encoded by GCT, GCC, GCA and GCG. The genes are the parts of the DNA that encode for genetic elements, and in particular for proteins.

The process of mapping from DNA sequences to proteins involves two major steps. The first step is the transcription of a gene into an RNA molecule, called a messenger RNA (mRNA). The second step is the translation of an mRNA into a polypeptide. The translation process depends on the presence of **transfer-RNA (tRNA)** molecules (see Figure 2.1) that make the mapping from codons in the mRNA to amino acids. Other RNAs, called **ribosomal RNAs** are involved in the translation machinery. In this dissertation, we are interested by the secondary structure of another genetic element: the RNase P RNA. It is a ribozyme responsible for the maturation of the 5' end of tRNAs [MJW98, LBK97].

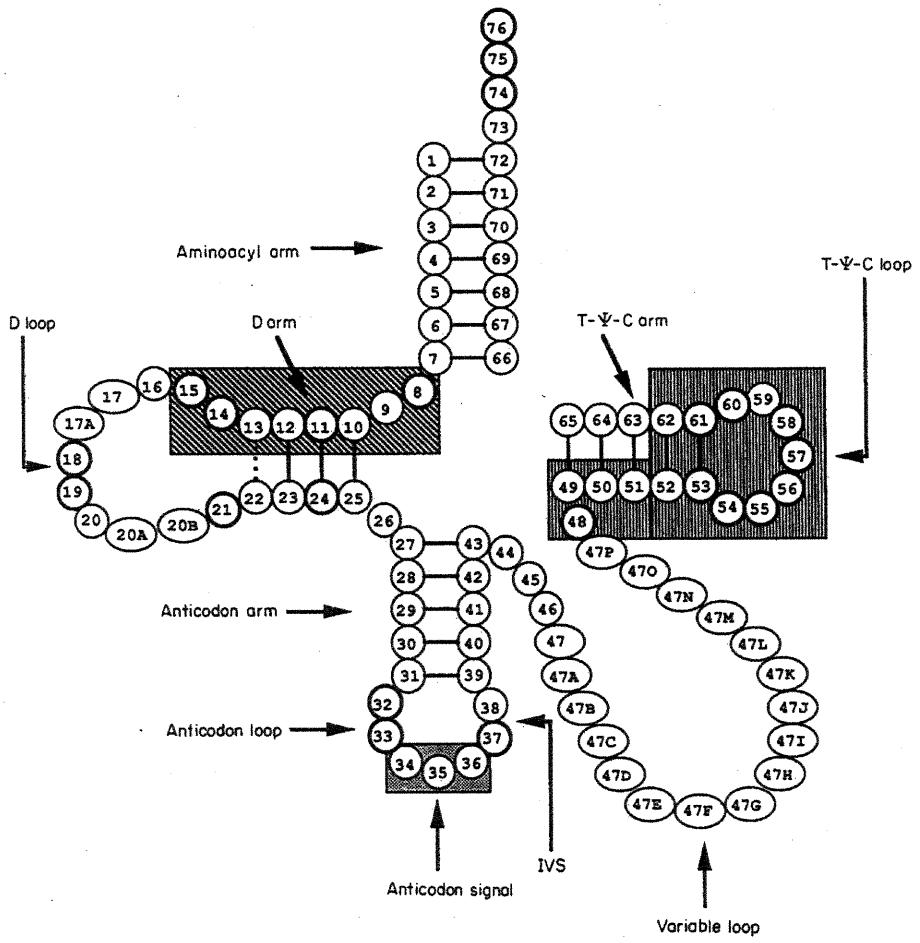


Figure 2.1 The cloverleaf shaped secondary structure of tRNA. The numbers are standard. The circles represent the nucleotides anytime present in the structure and the ovals which don't appear in every sequences of tRNA. The conserved bases are indicated by the circles in bold [EML96].

2.2 RNA secondary structures

An RNA is made up of a long chain of nucleotides (A,C,G,U). The base sequence that characterizes an RNA molecule is called its **primary structure**. Under natural

conditions, parts of a single RNA molecule bond to each other through complementarity, to define its **secondary structure** (see Figure 2.2).

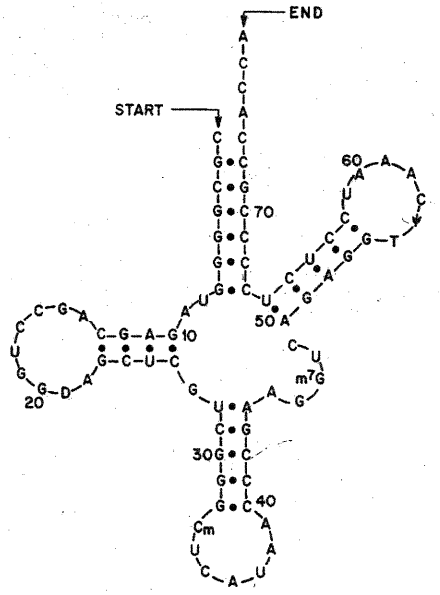


Figure 2.2 The secondary structure of a transfer RNA molecule, $tRNA_F^{Met}$ from Anacystis nidulans. Letters other than A, G, C, U indicate chemical modifications of these four unites [ECC76].

Secondary structures may have different and complicated shapes. However, any secondary structure S can be described in a unique and natural way as made up of different kind of paired and unpaired substructures (Figure 2.3) that we describe here. If $i \cdot j$ is a pair and $i< r <j$, we say that $i \cdot j$ surrounds r. Similarly, $i \cdot j$ surrounds a pair $p \cdot q$ if it surrounds both p and q.

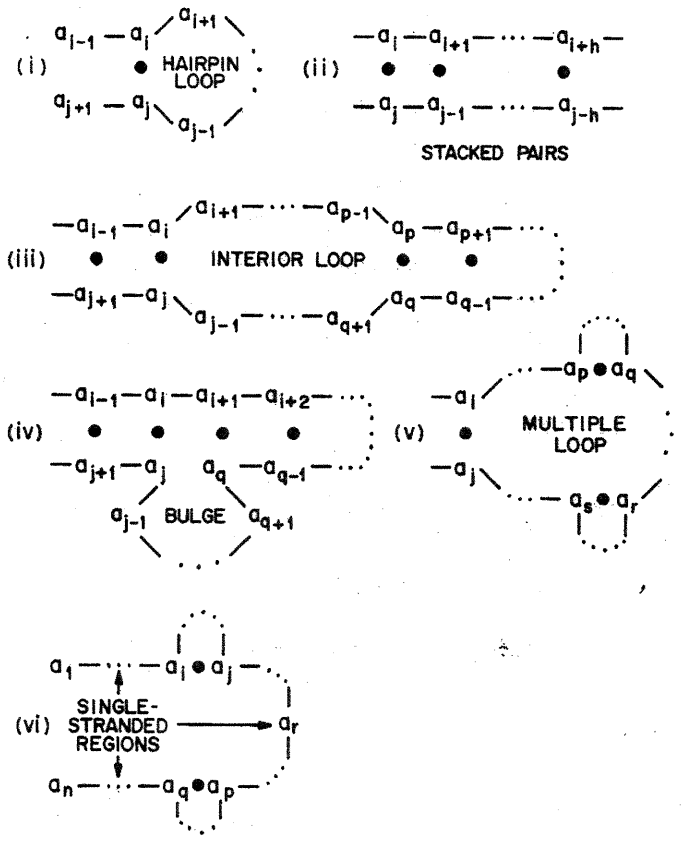


Figure 2.3 The six substructures: (i) hairpin loop (ii) stacked pairs (iii) interior loop (iv) bulge (v) multiple loop (vi) single-stranded regions [SK83].

1. If S contains i · j but none of the surrounded elements i + 1, …, j-1 are paired, the loop thus formed is called a hairpin.
2. If S contains i · j, (i+1) · (j-1), …, (i + h) · (j-h), each of these pairs (except the last) is said to stack on the following pair. Two consecutive pairs may be referred to as a stacked pair or as a stacked-pair cycle.

3. If i+1<p<q<j-1 and S contains i · j and p · q, but the elements between i and p are unpaired and the elements between q and j are unpaired, then the two unpaired regions are said to constitute an interior loop.
4. If S contains i · j and (i+1) · q, and there are some unpaired elements between q and j, these unpaired elements form a bulge. Symmetrically, a bulge also occurs if S contains i · j, p · (j-1) and some unpaired elements between i and p.
5. If S contains i · j and i · j surrounds two or more pairs p · q, r · s, ... which do not surround one another, then a multiple loop is formed.
6. If r is unpaired and there is no pair in S surrounding r then we say that r is in a (external) single-stranded region.

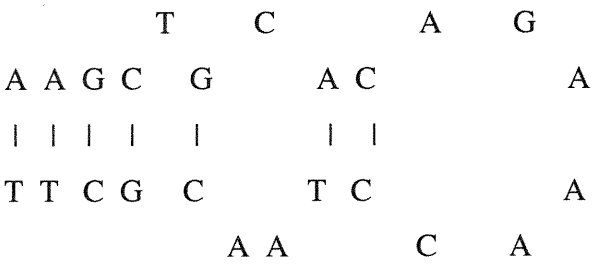


Figure 2.4 An example of a helix. The helix contains a loop of 6 nucleotides, a stem of stacked 7 pairs, a bulge (unpaired nucleotides at one side of the stem) and an interior loop (unpaired nucleotides at both sides of the stem).

In this dissertation, a **stem** will be any sequence of stacked pairs, bulges and internal loops, and a **helix** will refer to a stem followed by a loop. In the literature, this kind of structure is sometimes referred to as a stem-loop or a palindrom. Figure 2.4 is an example of a helix with a bulge and an internal loop.

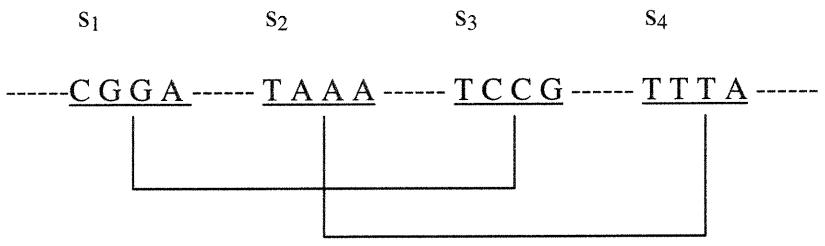


Figure 2.5 An example of a pseudoknot. s_1 and s_3 form a stem of stacked 4 pairs, s_2 and s_4 form another stem of stacked 4 pairs.

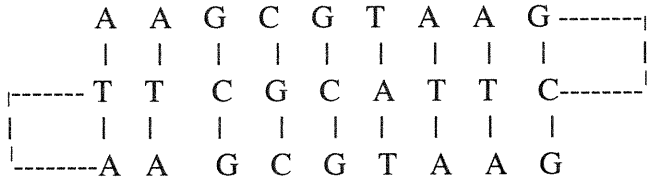


Figure 2.6 An example of a triple helix. It contains two loops and 9 base triples.

RNA secondary structures subsequently fold into tertiary structures. Pseudoknots and triple helices are part of such tertiary structures. A **pseudoknot** (see Figure 2.5) in a folded RNA molecule has a stem-loop plus a single strand folded back to form base pairs with the bases in the loop. RNA pseudoknots are functionally important in several known RNAs. For example comparative analysis shows that RNA pseudoknots are conserved in ribosomal RNAs, the catalytic core of group I introns, and RNase P RNAs [Cec93]. A base triple is an approximately planar group of three

bases involving at least one hydrogen bond joining each pair. A **triple helix** is a contiguous series of base triples (see Figure 2.6).

2.3 Multiple sequence alignment

Multiple sequence alignment is an important tool in studying RNA genes. The basic information they provide is the position and nature of the conserved regions in each member of the gene group. This is very useful in predicting the function and structure of RNA gene, and in identifying new members of gene families. Conserved sequence regions correspond to functionally and structurally important parts of the motif. We often only know the sequence-to-function relation for one or two members of the group. Multiple alignments let us transfer that knowledge to the other members in the group. Hypotheses about functional importance or specific roles can then be directly tested by mutagenesis and truncation experiments.

Let A be an alignment of n homologous sequences corresponding to a given region of an RNA family (for example $T\Psi C$ or D region of tRNAs). From such an alignment, one should be able to extract some common characteristics of the primary and secondary structure of these sequences, that is deduce a **consensus secondary structure**. Different methods have been used to represent common primary constraints. First, from an alignment A, we can deduce a **consensus sequence** that is the sequence formed by the most frequent nucleotide at each position of the alignment. For example, consider the following alignment:

A T A G T A C
A C A G A _ C
T T C G A A _

Its corresponding consensus sequence is: ATAGAAC, as A is the most frequent nucleotide at position 1, T is the most frequent nucleotide at position 2 etc.

For a more precise representation, we can also consider the **consensus matrix** representing the alignment, which is the matrix containing the frequency of each nucleotide at each position of the alignment. The consensus matrix representing the alignment above is given in Table 2.1.

position	Base			
	A	C	G	T
1	0.67	0	0	0.33
2	0	0.33	0	0.67
3	0.67	0.33	0	0
4	0	0	1	0
5	0.67	0	0	0.33
6	0.67	0	0	0
7	0	0.67	0	0

Table 2.1 The consensus matrix representing the alignment in the example.

For a real example, let us consider the $T\Psi C$ region of the tRNAs (between positions 48 and 62). El-Mabrouk and Lisacek [EML96] have aligned 546 RNA sequences extracted from the EMBL database, and obtained the consensus matrix shown in Table 2.2 for the $T\Psi C$ region.

Position	*Base*			
	A	C	G	T
48	0.013	0.833	0.005	0.148
49	0.147	0.303	0.514	0.037
50	0.106	0.473	0.216	0.205
51	0.161	0.158	0.520	0.161
52	0.126	0.013	0.835	0.026
53	0.009	0.000	**0.991**	0.000
54	0.039	0.000	0.000	**0.961**
55	0.000	0.002	0.002	**0.996**
56	0.000	**0.995**	0.004	0.002
57	0.299	0.000	0.701	0.000
58	**0.996**	0.000	0.004	0.000
59	0.476	0.038	0.214	0.271
60	0.050	0.197	0.006	0.747
61	0.000	**0.987**	0.002	0.011
62	0.026	0.826	0.013	0.136

Table 2.2 The consensus matrix for the $T\Psi C$ region.

From a consensus matrix, one can deduce a certain number of conserved nucleotides. A given nucleotide X (X is A, C, G or T) is said **conserved** at position p if X is the nucleotide present at position p in almost all the sequences of $\boldsymbol{A}$. For example, if we take as conserved nucleotides those that have a frequency higher than 0.9, then we can define the $T\Psi C$ region by the sequence: NNNNNGTTCNANNCN, where N corresponds to any nucleotide.

Suppose now that, not only we know the alignment, but also the secondary folding of the sequences of **A**. Suppose that the conserved nucleotide X is situated in a folded region, that is, it is part of a base-pairing X-Y. Then this pairing is called a **conserved base-pairing**. For example, the first conserved nucleotide of the $T\Psi C$ region is one part of a base-pairing G-C (see Figure 2.1, 2.2).

Chapter 3

Algorithms for searching structured biological motifs

3.1 Introduction

Just as the evolutionary relationship in proteins is often seen more in tertiary structure than primary sequence, RNA molecule relatedness is often seen in preserved secondary structures. Indeed, RNA secondary structure gives useful information about the mechanisms of gene expression, gene evolution and the functions of ribosome. Much effort has been devoted to finding an RNA structure given an RNA sequence. Phylogenetic analysis of homologous RNA sequences identifies secondary structures that are conserved during evolution [FW75] [WGG83] [JOP89]. Another approach is to apply thermodynamics to compare the free energy of alternative structures [TUL71] [NJ80] [ZS81] [JGS84]. The retained secondary structure is the one that has the lowest free energy value. Context-free grammars have also been applied to the problem of predicting the secondary structures of RNA families [Sea93] [ED94] [SBM94].

Following the complete or partial sequencing of a large variety of genomes, and in particular the human genome, one of the major challenges in molecular biology is to decode this huge amount of information by identifying the different genes in the new sequenced genomes. Given a gene family characterized by a particular secondary

structure, the searching problem is to identify, in a newly sequenced genome, all sub-sequences that are coding for a gene of the given gene family, i.e., all sub-sequences that can fold in a given way. Though less studied than the prediction problem, the specific problem of searching biological structures has been treated in different ways. Some methods are tailor-made for searching specific families, for example tRNAscan [FB91], tRNAscanSE[LE97] and FAStRNA [EML96] for tRNAs, CITRON [LDM94] for group I introns, and SNOSCAN for snoRNAs [LE99,OLR00]. Other methods are more general in the sense that they are not restricted to the identification of specific gene families, such as RNAMOT [GMC90], RNABOB[Edd96], Palingol [BKV96]. In Section 3.2, we describe two examples of tailor-made methods for tRNA identification. In Section 3.3, we describe RNAMOT, RNABOB and Palingol.

Most of the methods mentioned above do not rely on deep algorithmic considerations, which make them slow in practice. Another drawback is the lack of flexibility in defining the conserved sub-structures. In Section 3.4, we describe a new approach that will allow searching for biological structures in a very flexible way. This approach is based on partitioning the structure into conserved primary and secondary sub-structures, searching for these sub-structures, and then assembling them to form the final general structure.

Whatever the method is, it is always based on the identification of various conserved primary and secondary sub-structures. A flexible representation of primary structures is provided by regular expressions. In Section 3.4, we introduce some algorithms for the identification of all approximate occurrences of a regular expression in a text (or a genome). For RNA molecules, true signals are actually defined by a combination of spatial structure and sequence motif, and folding constraints can be stronger than the primary sequence itself. Few algorithms have been devoted to the search of conserved motifs with folding constraints, one of the most interesting one being the

Sagot-Viari algorithm [SV97] for searching helices, mirror-repeats and pseudo-knots. The major work of this thesis is to generalize this algorithm to a large variety of helices. These improvements will be presented in the next chapter, but in this chapter (Section 3.5), we describe in detail the basic Sagot-Viari algorithm.

3.2 A tailor-made algorithm for searching tRNA sequences

FAStRNA:

The tRNA molecule has been extensively studied. The corresponding gene sequence is short (about 75 nucleotides) and the primary, secondary and even tertiary structure is often conserved [RRB76]. Self-complementary regions create a cloverleaf-shaped structure (see Figure 2.1). This structure is subdivided into four regions: the aminoacyl stem, the *D* region, the anticodon stem and the $T\Psi C$ region. The aminoacyl stem (the acceptor stem) has 7 base-pairings, and a (external) single-stranded region containing 4 nucleotides. The anticodon has a stem of 5 base-pairings and a loop of 7 base-pairings with conserved nucleotides (nucleotide 33 is usually a T, and nucleotide 37 is usually an A). The $T\Psi C$ region has a stem of 5 base-pairings and a loop of 7 nucleotides. Finally, the *D* region has a stem of 3 or 4 base-pairings, and a loop of 7 to 11 nucleotides.

Several algorithms have been developed for identifying all tRNA genes in a genome [Sta80][PCB94][FB91][EML96][ED94][LE97]. One of the most accurate and fast algorithm is *FAStRNA*, which is a modified version of *tRNAscan* [FB91]. It is a backtracking algorithm based on the following considerations:

- The presence of invariant (or universal) nucleotides situated at specific positions.
- The cloverleaf structure consisting of four stems and three loops.

- A relatively optimized hierarchy of the operations. In other words, most constrained regions are searched before less constrained regions (the *TΨC* region is searched first, then the first part of the *D* region, then the aminoacyl stem, then the anticodon stem).
- The calculation of a general score for evaluating the stability of the entire structure.

By aligning about 500 tRNA sequences, N. El-Mabrouk and F. Lisacek have been able to represent the *TΨC* signal, situated between the nucleotides 48 and 62, by the consensus sequence: YMNNRGUUCRAKYCY, where each letter denotes a particular subset of {A, C, G, T}, formally defined in Table 3.1. The meaning of that consensus sequence is: the first position of the *TΨC* signal is either a C or a T in all the tRNA sequences of our test alignment, the second position is everything except a T, etc. Similarly, the *D* signal, situated between positions 8 and 15 has been represented by the consensus sequence: TRGYNNAR.

Symbol	Significance		Symbol	Significance
A	A		Q	A|T
C	C		R	A|G (purine)
G	G		S	A|C
J	C|G|T		T	T
K	A|G|T		W	G|T
L	A|C|T		Y	C|T
M	A|C|G		Z	C|G
N	A|C|G|T (whole alphabet)			

Table 3.1 Symbols used to define nucleotide subsets.

This representation of the *TΨC* and *D* signals consider the variability in the different tRNA sequences. Other flexibilities have been introduced to account for the *D*-loop variability and the dependencies between the different regions of a tRNA. Moreover, to improve the speed of the algorithm, efficient pattern-matching methods have been used to search for the *TΨC* and *D* signals. With these improvements, the previous (*tRNAscan*) algorithm was altered to run 500 times faster and to lower both rates of false positives and false negatives.

tRNAscan-SE:

tRNAscan-SE is another algorithm for identifying all tRNAs in a genome that has been developed by Todd Lowe and Sean Eddy [LE97]. It is an improved tool based on three previous methods: tRNAscan [FB91], Cove analysis [ED94] and Pavesi's algorithm (EufindtRNA) [PCB94]. tRNAscan-SE does no tRNA detection itself, but instead combines the strengths of the three independent tRNA prediction programs by negotiating the flow of information between them, performing a limited amount of post-processing, and outputting the results in one of several formats.

TRNAscan-SE combines the specificity of the Cove probabilistic RNA prediction package [ED94] with the speed and sensitivity of tRNAscan 1.3 [FB91] plus an implementation of an algorithm described by Pavesi and colleagues [FCB94], which searches for eukaryotic pol III tRNA promoters (the implementation referred to as EufindtRNA). tRNAscan and EufindtRNA are used as first-pass pre-filters to identify "candidate" tRNA regions of the sequence. These sub-sequences are then passed to Cove for further analysis, and output if Cove confirms the initial tRNA prediction. In this way, tRNAscan-SE attains the best of both worlds: (1) a false positive rate equally low to using Cove analysis, (2) the combined sensitivities of tRNAscan and EufindtRNA, and (3) faster search than that of Cove analysis and the original tRNAscan.

3.3 General methods

3.3.1 RNAMOT

Gautheret-Major-Cedergren [GMC90] have presented a general method for representing an RNA structure given a certain number of structural elements and primary constraints. The considered structural elements are stems and loops (unpaired regions). The idea is to describe an RNA structure by a list of its structural elements, each element followed by its position and length. The descriptor can also indicate the presence of conserved nucleotides, and the maximal number of mismatches allowed on conserved nucleotides. For example, tRNA structures can be represented by the descriptor shown in Figure 3.1.

In Figure 3.1, the first line is a list of all the structural elements (Figure 3.1 a). 's' is for single-stranded (unpaired) region and 'H' for helical region. The first line is followed by a description of the properties of each structural element, i.e., for helices (H), the minimal length, the maximal length, the number of errors allowed (pairs other than Watson-Crick), and finally any primary sequence constraint (Figure 3.1 b and c). The same parameters are used for single-stranded regions, except for errors (Figure 3.1 d and e). The last two lines of the descriptor are optional declarations. The first line indicates the order in which the elements should be searched (Figure 3.1 f), and the second line the total number of mismatches allowed on the conserved nucleotides (Figure 3.1 g).

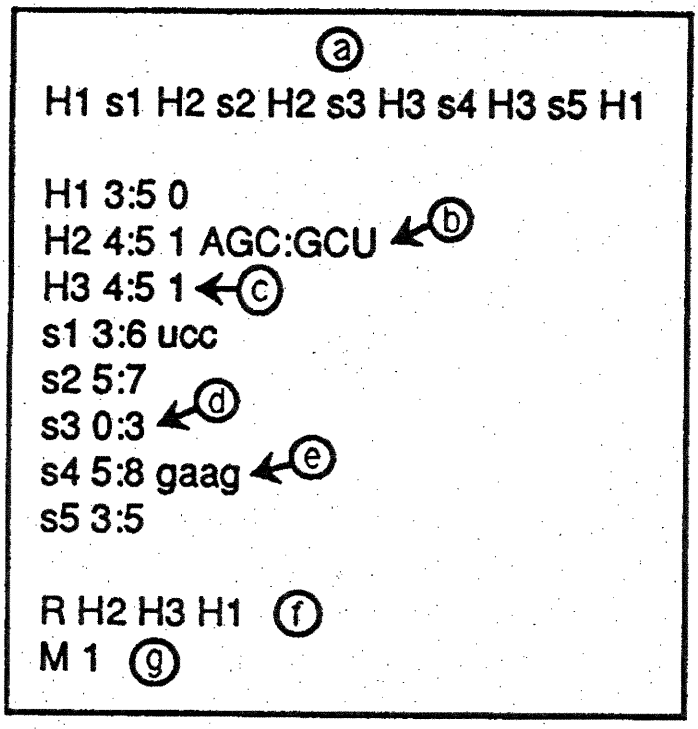


Figure 3.1 An example of descriptor. The organization of structural elements, single-stranded ('s') and helical ('H') regions is described in the first line. For each structural element, the constraints are given in the following order: minimal length, maximal length, maximal number of base mismatches (for helices only), and an optional primary sequence. The line starting with a 'R' is a search order command, the line starting with an 'M' gives the total number of base mismatches allowed within the motif.

RNAMOT is a backtracking algorithm that successively searches each structural element given in the descriptor. The order of searching can be modified by the user. When several matches are found at the same position, additional conditions on helix length and stability are used to choose the best match.

The advantages of RNAMOT are that it uses a simple representation of an RNA structure, it considers both primary and secondary constraints, it allows for a certain flexibility in the definition of the helices (a certain number of non Watson-Crick pairings is allowed), and a score for evaluating the stability of the global structure is calculated. However, the method is not flexible enough in defining primary structures constraints, as well as helix constraints. For example, motifs such as those defined in Section 3.1 with symbols from Table 3.1 cannot be considered. Also the only accepted errors are mismatches, thus interior loops and bulges are not allowed. As mentioned in the original paper [GMC90], its main drawback appears when complicated correlations between sequence and/or structure elements should be introduced.

3.3.2 RNABOB

RNABOB is an improvement of RNAMOT [Edd96]. It allows for non-canonical base-pairings, and for mismatches in the stacked regions as well as in the single stranded regions. The descriptor defines the order of occurrences of a series of single-stranded, double-stranded, and related elements. Each element is prefixed with 's', 'h', or 'r', indicating single-strand, helical, or a relational element. Helical and relational elements are paired to other elements, which are suffixed by a prime. For example, [h1 s1 h1'] describes a hairpin structure with a stem (h1 – h1') and a loop (s1). The relational elements are used for non-canonical base-pairings. For example, if the stem always contains a non-canonical base-pairing, the topology could be described as [h1 r1 h2 s1 h2' r1' h1'].

The structural elements are described as in the RNAMOT descriptor, except that relational elements have an additional field, a "transformation matrix" of four nucleotides, specifying the rule for making the r' pattern from the r sequence in order A-C-G-T. For example, the transformation matrix for a simple helix is TGCA; if you

consider G-T pairs, it is TGYR. RNABOB consider G-T pairing by default and uses the TGYR matrix for helical elements.

RNABOB is more flexible than RNAMOT as the relational elements consider all kinds of base-pairings. However, neither of these algorithms deals with insertions/deletions. Therefore, only very specific interior loops can be considered, those that have the same number of nucleotides at each side of the stem.

3.3.3 Palingol

Billoud-Kontic-Viari [BKV96] described a general representation of structures, together with a programming language, Palingol, designed to manipulate them. Palingol has specific data types, corresponding to structural elements, basically helices that can be arranged in any way to form a complex structure.

The general idea of the method is the following. At the beginning, the user should describe the structure as a list of helices and a list of two kinds of constraints: local constraints, that act on each individual helix specifying its length, the size of the loop, the presence of particular primary constraints; and global constraints that act between helices, specifying their relative position or any kind of cross-conditions and correlation between properties of different helices. Once all the local and global constraints are identified and written down in natural language, they are translated to a Palingol program. The rest of the analysis proceeds in two main steps: the search for elementary helices and the Palingol interpretation/search.

In the first step, the sequence is scanned by an internal program (HelixSearch) which builds, for each sequence, a database of all "elementary" helices found in the sequence, that could be involved in the final structure. HelixSearch can treat non-canonical base-pairings, but not bulges. Notice that this program can be replaced by another one that is more flexible in defining basic helices.

The second step is performed by the Palingol interpret and engine. The interpreter reads the user's program written in Palingol, and builds an evaluation tree for all the constraints. Then the engine runs through the list of helices, trying to find all subsets of helices that match the required constraints. This is done by a branch-and-bound algorithm.

The elementary objects manipulated by Palingol are the helices computed by HelixSearch. Each helix is described within Palingol by three physical elements, respectively called: 'head', 'tail' and 'loop' where 'head' and 'tail' represent the two paired regions and 'loop' represents the region in between. The start and end positions of each of these three elements on the sequence are respectively referred to as 'start' and 'end'. A real secondary structure is actually described by the association of several elementary helices. More precisely, it is described by a set of elementary helices (each of them with local constraints) and a set of constraints between them.

The advantages of Palingol is that it allows for searching all kinds of complex helices, with bulges, interior loops, non-canonical base-pairings, and it allows for mismatches in double stranded regions, as well as single stranded regions. However it still does not allow for insertion/deletion errors. But the major drawback of the method is that a relatively simple secondary structure requires a very complex and tedious representation in Palingol language. Moreover, even if bulges can be considered, they should be specified very precisely, and no flexibility in their length or position can be considered without highly increasing the complexity of the representation. This impractical representation make it very difficult to search for complex biological structures such as those introduced in Chapter 4, and to try different variations of the same general structure.

3.4 Searching for conserved structures

As we have seen in the last section, all methods for searching complex RNA structures are based on the identification of various conserved primary and secondary sub-structures. RNAMOT and RNABOB require a unique subdivision of the general motif into structural elements. As for Palingol, it requires the use of a preliminary program "HelixSearch" that generates a database of all "basic" helices that can be part of the general motif. The more flexible HelixSearch is, the easier the representation of the general structure is, and the more efficient Palingol can be. Indeed, if HelixSearch is restricted to elementary helices, without considering non-canonical base pairings, bulges or interior loops, then the general structure should be subdivided into many substructures, and the assembling procedure becomes very long.

Despite the drawbacks of Palingol, the general idea of pre-processing the genome to be analyzed and creating a database of the different kinds of helices before assembling them in one or different ways seems very promising, as it allows for a very flexible search. Indeed, in that way, the user can try to assemble the "basic" helices in different way, and new structures can be more easily discovered. This motivates the development of very efficient and flexible algorithms for the identification of conserved primary and secondary sub-structures.

3.4.1 Searching for primary structures

In many cases, conserved motifs are the anchor points for identification of complex biological structures. These motifs can be defined by sequences, such as "TAGCTCAG" or, in a more flexible way by subsets of nucleotides at each position, such as "TRGYNNAR". In Section 3.2, we have described the $T\Psi C$ and D signals of the tRNA by this kind of motifs. More flexible motifs can be defined by regular expressions. For example in the motif "AC (GG | TA) CT (GT)?", '|' represents the union operation, and '?' means one or zero times the preceding expression. More

complex motifs can also be defined by the combination of several regular expressions, separated by gaps.

The problem of approximately matching a sequence in a text has been extensively studied by various people in the pattern-matching field. In the case of biological motifs, the most interesting motifs are those represented by regular expressions. Myers and Miller [MM89] have developed an O(np) algorithm for approximately matching a sequence G of size n to a regular expression R of size p. This algorithm is based on an alignment graph obtained by concatenating n+1 copies of a non-deterministic finite automaton recognizing R. An optimization of this algorithm running in O(kp) time, where k is the number of allowed mismatches, has been developed by Myers [Mye96]. Myers has also considered the case of motifs formed by regular expressions interspersed with specifiable distance range. For example, if S1, S2, S3, S4 are four regular expressions, a motif can be specified by:

"{S1,1}<0,20>({S2,1}|{S3,1})<25>{S4,1}"

which represented the class of patterns specified by: << S1 with at most one error, followed at a maximum distance of 20 nucleotides by an S2 (with at most one error) or an S3, followed by an S4 at a distance of 25 nucleotides. For this class of patterns, Myers [Mye96] develops a backtracking procedure with optimal evaluation order in the sense that its expected time is minimal over all such procedures.

3.4.2 Searching for helices

Few algorithms have been devoted to the identification of conserved motifs with base-pair constraints. PatScan [OB00] is one of such algorithms. It is a pattern matcher which searches protein or nucleotide sequence archives for instances of an input pattern. It can also searches for palindrome or complementary sequences, and thus for simplified helices. The Sagot-Viari algorithm [SV97], which is described in

details in the next section, is a more efficient algorithm that allows for the approximate search for helices, palindromes, mirror repeats, pseudo-knots and triple helices.

Another interesting algorithm related to this problematic is the Gendron-Major [GGM98] algorithm. It identifies, in an RNA structure, the most represented helices of small size. The algorithm represents helices by graphs of relations where the nodes represent the nucleotides and the edges represent structural relations. The problem of finding helices in an RNA secondary structure can be divided into two distinct tasks. The first one is an enumeration of all possible sub-graphs and, second, their isomorphic classification. Central to the helix identification process is the notion of incremental enumeration. In order to find the sub-graphs of size n, the sub-graphs of size n-1 are considered. The sub-graphs of size n-1 are extended by connecting the nodes that are connected to it from the secondary structure. The classification of sub-graphs requires an efficient graph isomorphism algorithm. Specific RNA secondary structure information makes it possible to split the isomorphism determination in three stages of increasing complexity. First, a comparison is made between two sub-graph vertices, based on their respective type and number of relations, or degree. Then, if the sub-graphs contain the same nucleotides, their edges are compared, and if they are equal, a depth-first search is finally applied to verify their isomorphism.

3.5 Sagot-Viari Algorithm

Marie-France Sagot and Alain Viari [SV97] have developed algorithms for flexibly identifying structural objects in nucleic acid sequences. These objects are helices, mirror repeats, pseudoknots and triple helices. The developed algorithms are not predictive in the sense that we cannot say which of the potentially structural objects

found will actually be part of the final, global structure of the molecule, but rather identify all those that may do so.

As we are interested here in finding helices, we restrict ourselves to describe the Sagot-Viari method that concerns helices, though the methods for the other structural objects are not much different. More precisely, the problem solved by Sagot-Viari algorithm is the following: given 4 parameters d_{min}, d_{max}, k, e, find in a genomic sequence s, all the helices of maximal stem size k and with a loop size varying between d_{min} and d_{max} with at most e errors. The valid base-pairing are A-T and G-C, and all other base-pairings are considered as errors.

The algorithm treats the problem of errors with the help of an object called a model against which the comparisons are made. A model is a word over the alphabet of nucleotides corresponding to one half of the helix being searched. The second part of the helix corresponds only approximately to the model.

Next, we describe the method and the algorithm with the help of several examples.

3.5.1 Statement of the problem

Let Σ be the alphabet of nucleotides, that is, Σ = {A, C, G, T or U} and let a sequence s be an element of Σ^*. A word u of length k is an element of Σ^k for $k \geq 1$ and u is said to be a word in s if s = xuy with x, y $\in \Sigma^*$. Here is an example of a sequence s:

A	A	C	T	C	A	C	G	T	C	C	G	T	T	G	A	C	G	T	A	C	T	T	T	A	C	G	T	C	A	T
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Let u =*ACGTCA* be a word of size 5. There is an occurrence of u starting at position 25 and ending at position 30 in the sequence s. We say that 25 is the **start position**

of this occurrence of u in s and 30 is its **end position**. The **inverse** $\bar{u}$ of u is just the word read in reverse, that is $\bar{u}$=*ACTGCA*. 30 is the **start inverse position** of the occurrence of u in s and 25 is the **end inverse position of** this occurrence in s.

Let M_c be the 4×2 matrix of the nucleotides complementary base pairs:

M_c =

A	T(U)
C	G
G	C
T(U)	A

where (A, T) and (C, G) are the Watson-Crick base pairs. The **complementary inverse** u_c of u is the word obtained by reading u in reverse and replacing each nucleotide by its complementary according to matrix M_c. For example, for the word u given before, $u_c = T\,G\,A\,C\,G\,T$.

Notice that for any word u, if $v = u_c$, then $v_c = u$.

Definition 3.5.1 Given non-negative integers e, d_{min} *and* d_{max}, and two words u, v in s, we say that (u, v) forms an **approximate helix** in s if it satisfies the following constraints:

- $dist_L(u, \overline{v_c})$ is no more than e, where $dist_L(x, y)$ is the edit distance between x and y (it is the minimum number of substitutions, deletions and insertions necessary to convert x into y);

- $d_{min} \leq d \leq d_{max}$, where d is the distance between the end position of v and the start position of u in s.

If $e = 0$, we have of course an exact helix.

In the example given in Figure 3.2, if $e = 1$, $d_{min} = 2$, $d_{max} = 20$, we can say that (u, v) forms an approximate helix in s because $dist_L(u, \overline{v_c}) = 0, \leq e$, and $d = 25-19+1=7$, *i.e.* $d_{min} \leq d \leq d_{max}$. Moreover, (u, v) is an exact helix in s. There is another approximate helix (u, w) in s, with $w = T\ C\ A\ C\ G\ T$, because $dist_L(u, \overline{w_c}) = 1, \leq$ e, $d = 25-9+1=17$, i.e. $d_{min} \leq d \leq d_{max}$.

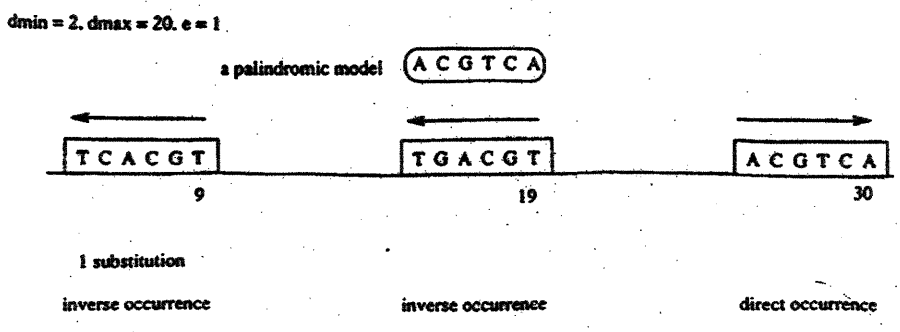


Figure 3.2 Example of palindromes and palindromic model.

Mismatches in a helix correspond to interior loops, and deletions and insertions to bulges. The subsequence of s between the two halves of a helix represents the helix loop (see Figure 3.3).

Figure 3.2 has also given a **palindromic model** $m = A\ C\ G\ T\ C\ A$. When the program searches for helices in the sequence s, the words in s should be compared with the model m. Furthermore, the words should be read in both directions as shown by the arrows in Figure 3.2. When the direction is from the beginning to the end of the sequence, it is searching for direct occurrence and the words should be exactly equal to the model m; when the direction is from the end to the beginning of the sequence, it is searching for complementary inverse occurrences and the words should be related with the model m with an upper bounded edit distance. In Figure

3.2, m is exactly equal to word u, so we say that u is an (exact) direct occurrence of m in s; v and w are two approximate complementary inverse occurrences of m, because $dist_L(m, \overline{v_c}) \leq e$, and $dist_L(m, \overline{w_c}) \leq e$. The set of all direct occurrences of a palindromic model m present in s is denoted by $OD(m)$ and its set of complementary inverse occurrences is denoted $OI(m)$.

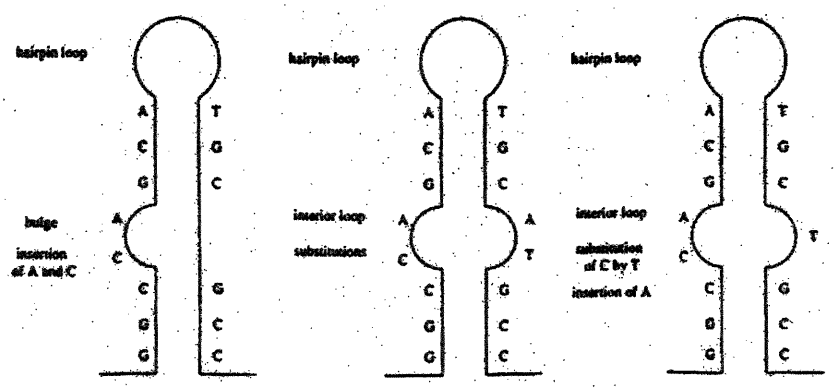


Figure 3.3 Mismatches and deletions/insertions corresponding to bulges and interior loops.

With palindromic model, it is not necessary to have $OD(m) \cap OI(m) = \varnothing$. For instance, palindromic model $m = AGCT$ is present in sequence s=$AGCTAGCTAGCT$ with the constraints (e=0, d_{min}=0, d_{max}=0) (at positions 5 and 9). The word $AGCT$ starting at position 5 is both a direct and a complementary inverse occurrence.

The parts of the structural helices searched for are not kept "assembled" in the sets of occurrences of a model, but are instead kept separated in $POD(m)$ and $POI(m)$ until the end. Here $POD(m)$ and $POI(m)$ are the sets of possible direct occurrence and possible inverse occurrence respectively. In the exact case for instance ($e = 0$), keeping them assembled would require $O(n^2)$ space in the worst case while keeping

them separated requires at most $2n$ space. Note however that, although the parts of a structure are kept apart, only valid ones are preserved. If a word u kept in $POD(m)$ is such that $u = m$ for a model m, u will be stocked in $OD(m)$ only if there exists v in $POI(m)$ such that (u, v) is a helix satisfying (e, d_{min}, d_{max}) constraints, and vice-versa. This is the algorithm **Verify** (See Figure 3.4). What is even more important to observe is that checking for this validity takes only $O(n)$ time. Now the problem can be stated in the following way:

The Helix Problem Given a sequence s and non-negative integers e, d_{min} and d_{max}, the problem is to find all approximate palindromic models present in s that satisfy the constraints (e, d_{min}, d_{max}).

3.5.2 Algorithm

The algorithm for constructing the models and searching for their occurrences (see Figure 3.4) is based on the observation that models, and their sets of occurrences, can be constructed by recurrence. The main idea of the algorithm of searching for helices is to progressively construct models having at least one direct occurrence and one approximate complementary inverse occurrence that form a helix verifying the constraints (e, d_{min}, d_{max}).

Observe one fact in the construction. A word that is an occurrence of a model is related not just to the model, but also to at least one other occurrence of the same model. The relation of an occurrence to a model is either an exact match or an upper bounded edit distance, while the relation to another occurrence is positional (it is the distance between the end position of an occurrence and the start position of the other).

En summarize, the different steps of the algorithm, that are detailed in **ConstructModel** are the following:

- Begin by a model m of size 1;

- Find all exact direct occurrences of *m*, put them in *POD(m)* and find all approximate complementary inverse occurrences of *m*, put them in *POI(m)*;

- For a model of size 1, the algorithm just scans the genomic sequence to find all the occurrence of the given character; For the elongated model ($m'=m\alpha$) of size greater than 1, it takes each position in the last obtained sets *OD(m)* and *OI(m)* and check in the sequence whether the occurrence can be extended by the new character α;

- Filter the sets *POD(m)* and *POI(m)* (Function **Verify**) to keep the occurrences in *POD(m)* that have a corresponding occurrence in *POI(m)*, and vice-versa. This function gives size to the sets *OD(m)* and *OI(m)*;

- At the end, the direct and indirect occurrences are combined to form helices;

This procedure is shown in the algorithm in Figure 3.4. For simplicity, we assume a fixed size h_{max} for the desired helices. There is a remind of the definitions in the Figure 3.4.

Definitions	
m	model to be searched
OD(m)	set of direct occurrences of the model m
OI(m)	set of complementary inverse occurrence of the model m
ODI(m)	OD(m)∪OI(m)
POD(m')	set of possible direct occurrences of the model m', m'=mα
POI(m')	set of possible complementary inverse occurrence of the model m'
PODI(m')	POD(m')∪POI(m')
StackError(m)	set of errors for helices corresponding to the occurrences contained in set ODI(m)
h_{max}	maximum length of the model
maxerror	maximum error allowed for the helices
seq	the sequence being analysed

```
program DoMoivre(seq, h_max, maxerror)
   /*initialisation*/
   k ← 0
   m ← ()
   ODI ← ()
   PODI ← ()
   ConstructModel (seq, m, h_max, maxerror, ODI, k, PODI)
   ConstructHelices (OD(m), OI(m))
```

```
/*The k^th iteration in constructing a model m'=mα is:*/
Algorithm ConstructModel ( seq, m, h_max, maxerror, ODI(m), k, PODI(m) )
1:  if ( k ≤ h_max )
2:     for α = A, C, G, T
3:        m' = mα  //  add α to the end of model
4:        //search for elongated words to get sets PODI(m')
          ManberSearch ( m', ODI(m), StackError(m), PODI(m'), StackError(m') )
5:        // filtering POD(m') and POI(m') to get sets OD(m') and OI(m')
          Verify (POD(m'), POI(m'), OD(m'), OI(m') )    //See 3.5.1
6:        ConstructModel (seq, m', h_max, maxerror, ODI(m'), k+1, PODI(m'))
7:        remove α from the end of the model
8:     end for
9:  end if
```

```
Algorithm ManberSearch (m', ODI(m), StackError(m), PODI(m'),
                        StackError(m'))
1:  while get the next elements w from ODI(m) and  e from StackError(m)
2:        extend w in sequence and get wβ
3:        if w is a direct occurrence
4:           if α = β  // where α is the last element of model m'
5:              put wβ in set PODI(m'=mα) and put e in set StackError(m')
6:        else                                  // w is a complementary inverse occurrence
7:           if α match β (A-T or G-C) // where α is the last element of model m'
8:              put wβ in set PODI(m'=mα)   and put e in set StackError(m')
9:           else  if e+1 ≤ maxerror,    e ← e+1
10:                 put wβ in set PODI(m'=mα)   and put e in set StackError(m')
12: end while
```

```
Function Verify ((POD(m'), POI(m'), OD(m'), OI(m') )
      Returns sets OD(m') and OI(m') after filtering POD(m') and POI(m')
```

Function **ConstructHelices** (OD(m'),OI(m'))
Returns combined helices after assembly the elements from OD(m') and OI(m')

Figure 3.4 Recursive algorithm for constructing the model and algorithm of searching for the occurrences of model m'=mα.

3.6 Complexity

Algorithms for finding helices follow in general a naïve approach [Wat89], or restrict themselves to exact comparisons [Kon93] [Mar83]. In contrast, the Sagot-Viari algorithm is flexible and reasonably efficient in time and space. One of the reasons of this efficiency is that the left and right parts of the helices are assembled only at the end of the algorithm. Thus, the complexity of the algorithm depends on the number of parts that compose the helices, rather than on all possible helices contained in the considered genomic sequence. In the case of helices without errors, an upper bound for their number is $O(n^2)$ where n is the length of the genome, while the total number of the parts composing them is bounded over by O(n). Where errors are allowed, the Sagot-Viari algorithm finding all the parts composing the helices have time in $O(nk(e+1)(1+ min\{(d_{max} - d_{min} + 1 + e), k^e |\Sigma|^e\}))$, where n is the size of the genomic sequence, d_{min}, d_{max}, e, k are the parameters described above, and $|\Sigma|$ is the size of the alphabet of nucleotides. Putting the parts together requires $O(N)$ time, where N is the number of possible helices and is majored by $O(n(d_{max} - d_{min} + 1))$.

Chapter 4

Identifying helices in a genome

4.1 Introduction

Several methods have been developed for identifying more or less complex RNA structures. Whatever the method is, it is always based on the identification of various conserved primary and secondary sub-structures. In this section, we are exclusively concerned with secondary sub-structures. More precisely, we are interested in developing a flexible and efficient method for identifying all occurrences of specific secondary structures in a genomic sequence *G*. These secondary structures are helices defined by their stem and loop length, the presence of conserved bases, and the presence of interior loops and bulges.

We have presented in the section 3.5 the efficient Sagot-Viari algorithm for identifying helices. However, this algorithm searches for very general helices only described by their stem and loop length. In most cases different helices of a complex structure differ not only in their length, but also in some conserved bases, internal loops and bulges. It is important to introduce this information in the algorithm. Moreover, the score calculation should account for these different structure and sequence constraints. The algorithm is further improved through filtering the solutions and selecting the most significant occurrences. With these improvements,

our new algorithm gives a very flexible representation and identification of all kinds of helices.

In a first step, we study the different kinds of helices that can be encountered in complex structured biological motifs in order to better understand the needs and to perform the right improvements. After this preliminary study, we describe our new algorithm.

4.2 Conserved helices

Some common secondary structures of certain RNA families and other genetic elements have been determined by using different empirical methods, multiple sequence alignment and dynamic programming algorithms. The complex consensus structures contain different kinds of helices.

Before being able to develop an efficient algorithm that could be used by biologists in different situations and for different gene families, we should study the different helices that are encountered in biological data. This section is an overview of the different kinds of helices found in different consensus structures.

4.2.1 Conserved structures in tRNAs

As described in Section 3.2, the tRNA molecule has a very constrained cloverleaf structure (see Figure 2.1). In this structure, the $T\Psi C$ and D regions are very constrained. El-Mabrouk and F. Lisacek [EML96] have been able to represent the $T\Psi C$ region by the consensus sequence: YMNNRGUUCRAKYCY, where each letter denotes a particular subset of {A, C, G, T}, formally defined in Table 3.1. In order to take the secondary structure into account, we can represent the $T\Psi C$ region by the structure represented in the right of Figure 4.1. The left structure in Figure 4.1 is a possible occurrence of the right structure.

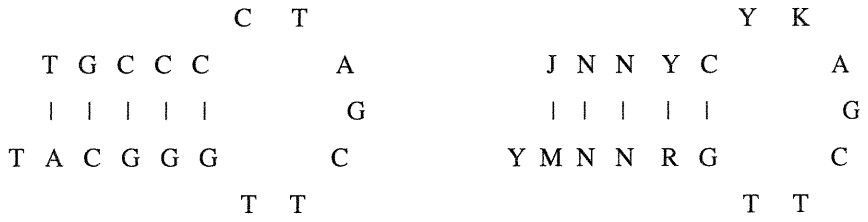


Figure 4.1 The right structure is a secondary expression representing a consensus for the *TΨC* region of tRNAs. The left structure is an occurrence of the right structure.

El-Mabrouk and Lisacek [EML96] also deduce a consensus structure for the *D*-region, represented in Figure 4.2. In this figure the symbol `?' means that the considered nucleotide can be missing.

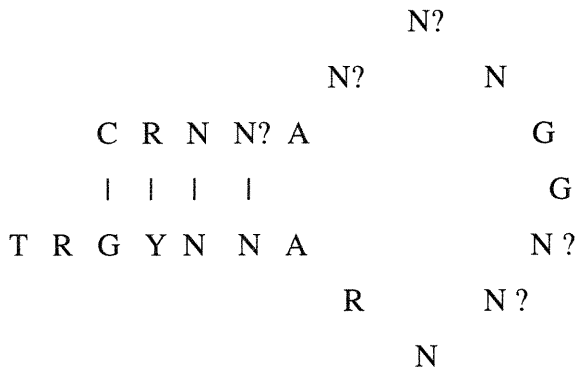


Figure 4.2 A consensus structure for the *D*-region of tRNAs. The notation '?' means that the nucleotide can be present or absent.

4.2.2 Conserved structures in mitochondrial 5S rRNAs

The most common configuration of RNA structures is alternating sections of helices and loops, called a helix-loop or loop-helix motifs. Also common in RNAs are mispaired bases, called noncanonical pairings. The most stable non-canonical base-pairing is *G-T*. The mitochondrial 5S rRNA in Figure 4.3 shows us this kind of common configuration of RNAs.

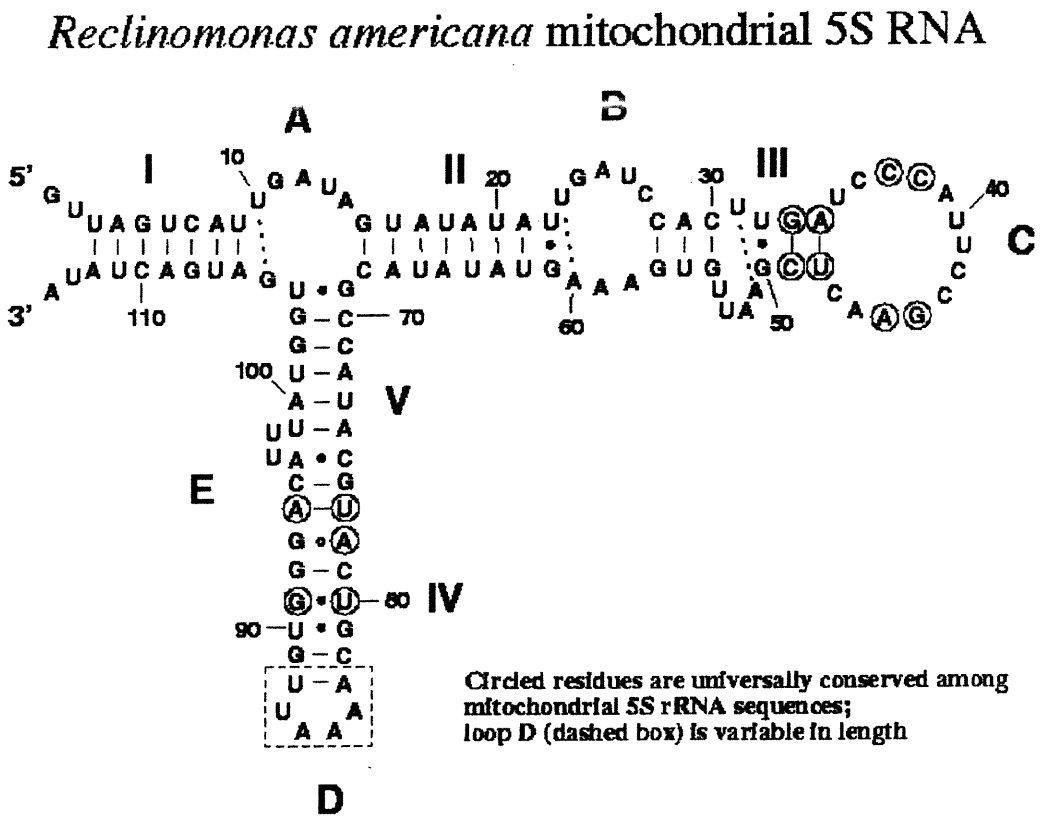


Figure 4.3 The secondary structure of Reclinomonas Americana (AF007261 – entry name in GenBank) mitochondrial 5S rRNA. Circled residues are universally conserved among mitochondrial 5S rRNA sequences. Loop D (dashed box) is variable in length.

Mitochondrial 5S rRNA molecules vary in length between 110 and 125 nucleotides, with the majority measuring about 120 nucleotides. The consensus structure can be subdivided into five helices: I, II, III, IV and V, and 5 loops: A, B, C, D and E. Although the structure is not as well conserved as that of tRNAs, some general characteristics have been deduced from the study of a set of aligned sequences.

The length of helix I vary between 6 and 10 base-pairings. The helix II is of size 8, but it often contains non-canonical base-pairings. Up to 4 non-canonical base-pairings have been observed.

The hairpin structure formed by helix III and loop C is the most constrained region. It contains 7 conserved nucleotides: 4 in the helix and 4 in the loop. The loop is of size 13. The helix contains 6 base-pairings, and a well-conserved interior loop located in the middle of the stem. The 5' part of the interior loop contains one nucleotide, whereas the 3' part contains between 3 and 7 nucleotides. By considering all the structures available, we have been able to represent it by the consensus structure of Figure 4.4.

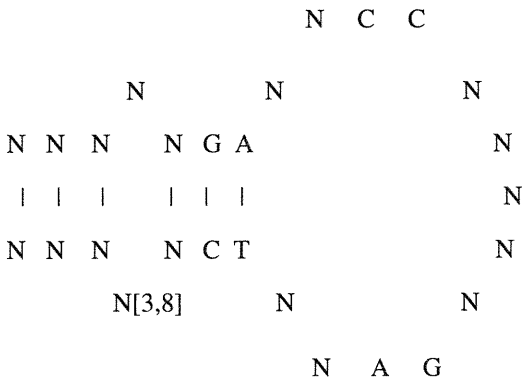


Figure 4.4 A consensus structure for the helix III in 5S rRNA.

Finally, the hairpin structure formed by helices IV and V and loops E and D is the one that contains the most non-canonical base-pairings. It has 14 base-pairings, among which 3-8 are non-canonical. Moreover, a bulge of variable size is located in the middle of the stem. Five conserved bases are present, four of them form two canonical base-pairings, and the last base is part of a non-canonical base-pairing.

4.2.3 Conserved structure in mitochondrial RNase P RNA

RNase P RNA is a ribozyme responsible for the maturation of the 5' end of tRNAs and various ribosomal RNAs. Its sequence (see Figure 4.5) varies in length between 160 and 900 nucleotides. Such molecules have been identified in 15 mitochondrial genomes. Though very different, the structures found have common characteristics, and the consensus sequence is formed of 19 regions that are called helix structures (P1, P2, ...P19). Every structure found is constituted of a subset of these 19 helices, but does not necessarily contain them all. Some structures contain just three of these helices: P1, P4 and P18, with several loop sections and single-stranded regions. The helices P1, P4 and P18 are the only ones that are located in all the structures.

The helix P4 is the most constrained region. By considering all the structures available, we have been able to represent it by the structure of Figure 4.6. Notice that conserved bases are located not only in the paired region, but also before and after this region. The P4 loop can be very large (almost the size of the whole RNase P RNA sequence).

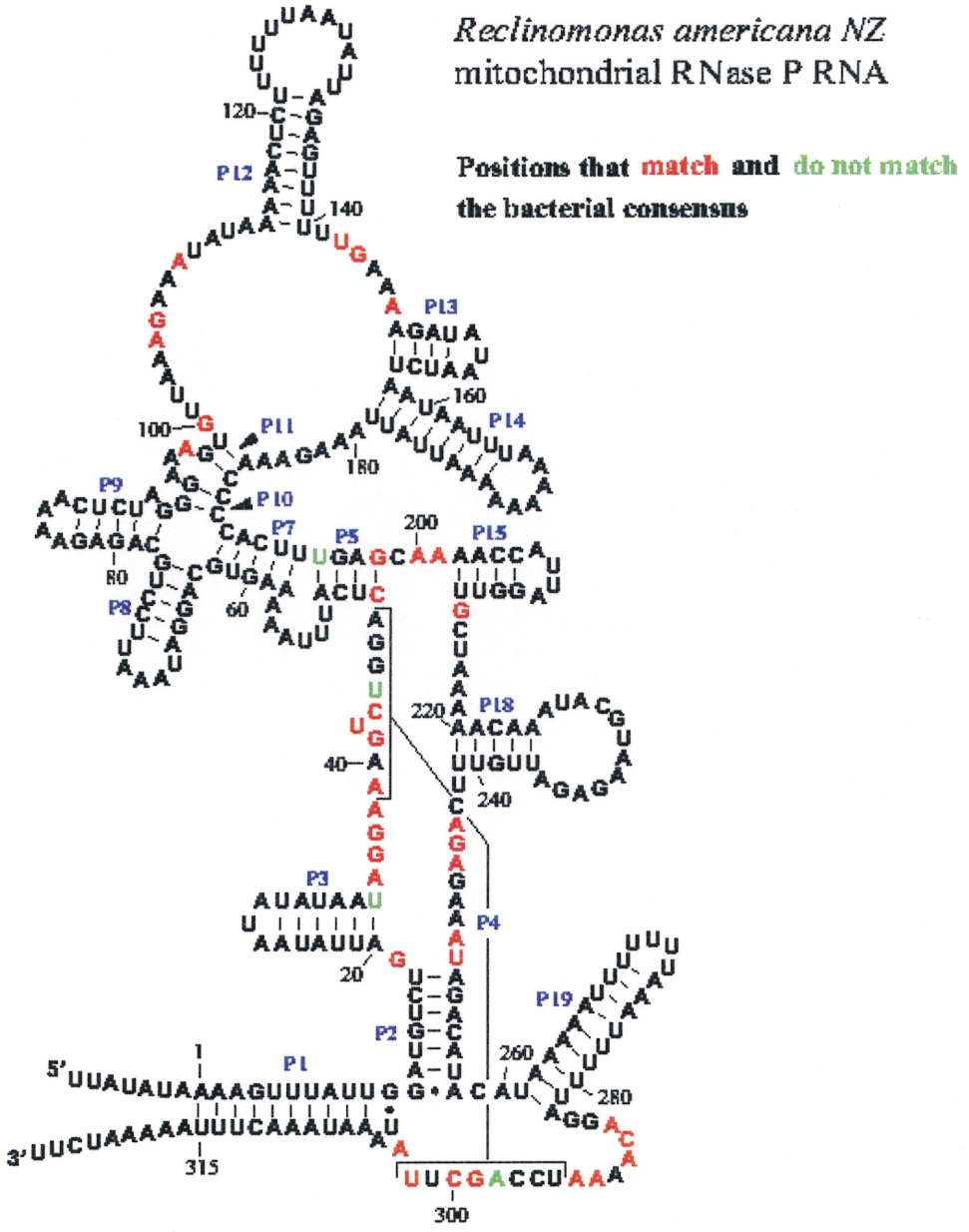


Figure 4.5 Secondary structure of reclinomonase Americana NZ (AF007261 – entry name in GenBank) mitochondrial RNase P RNA.

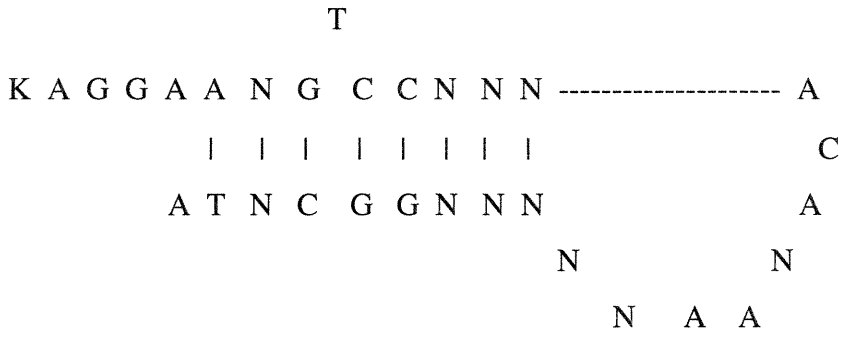


Figure 4.6 A consensus structure for the P4 region of the RNase P RNA. The line "-------" denotes a large loop .

P1 is the longest helix of the structure: its size varies between 11 and 27 nucleotides. In some cases, a bulge or interior loop is observed. P18 is a hairpin structure with a variable stem length (between 3 and 15 nucleotides) and a variable loop length (between 4 and 17 nucleotides).

Finally, we can notice the remarkable conserved multiple-loop structure formed by the helices P7, P8, P9 and P10.

4.3 Our algorithm for searching helices

The Sagot-Viari algorithm is improved in different ways. First, in order to avoid the redundancy in the solutions obtained by the algorithm, we introduce helix scores that reflect helix stability, and the solutions are filtered depending on their scores. This filtering largely reduces the number of irrelevant solutions. Furthermore, we consider the G-T pair as a valid one, but we give it a lower score than to Watson-Crick base-pairings. The helices together with their corresponding scores are reported as output.

Our second major improvement concerns the different kinds of helices that the Sagot-Viari algorithm is not able to search for. Indeed, in most cases, we not only know the length of a helix stem and loop, but we also have information about conserved nucleotides, bulges and internal loops that are present at specific position (see for example helices in Figures 4.2, 4.3, 4.4, 4.5, 4.6). Our goal is to introduce, in the Sagot-Viari algorithm, different constraints on the primary and secondary structure, in order to be able to search for various kinds of helices and to represent them in a very flexible way.

4.3.1 Filtering the solutions

The solutions obtained by the Sagot-Viari algorithm contain a high degree of redundancy. Indeed, when an occurrence of a helix is found at one position in the genome, many other similar helices are found in the same region. Usually, only one of these occurrences is significant. In order to choose the helix that has the highest stability of the group, we introduce scores that measure helix stability.

	A	C	G	T	E
A	-2	-2	-2	5	-2
C	-2	-2	7	-2	-2
G	-2	7	-2	3	-2
T	5	-2	3	-2	-2
E	-2	-2	-2	-2	

Table 4.1 the matrix that shows the score calculation rule.

Table 4.1 gives the scores used to evaluate each alignment between two nucleotides. E denotes the empty character, so that an alignment E-N or N-E, where N is an

arbitrary nucleotide, represents an insertion or a deletion of the nucleotide N. This scoring has been chosen to reflect the fact that the pair G-C is more stable than the pair A-T who is, in turn, more stable than the pair G-T [Tur88]. All other pairs of nucleotides are considered as mismatched and given the score -2. Similarly, an insertion or deletion (in a bulge or interior loop) is given the score -2.

Observe that the introduction of scores is not for replacing the notion of errors and uncertainty. We still preserve the way to count the errors in the word searched for in order to decide whether we continue or stop extending the word. The score role is just to filter the output solutions at the end of the algorithm. First, we introduce a parameter, *minscore*, that serves as a threshold. In other words, helices that have scores less than *minscore* are eliminated. The second filtering is to choose the best helix representative between several helices that have tiny position differences, which will be explained with an example.

Let *s* be the sequence:

$$s = \underset{1}{\text{GGGGGGAAAG}}\text{------}\underset{10}{\text{CTTTGCCCC}}\underset{17}{}\underset{25}{}$$

Let *m*=TTTGCCCC be the palindromic model searched for, *errmax* = 1 be the maximal number of allowed errors, and the 7-9 be minimal and maximal length of the loop respectively. The Sagot-Viari algorithm outputs three solutions satisfying these constraints:

Solution 1, beginning at position 2 and ending at position 25 in s:

```
G G G G G A A A ------|
| |  | |  •  | |  |     |
C C  C C G T  T  T ------|
```

Its loop length is 8, its number of errors is 1, and its score is 41.

Solution 2, beginning at position 1 and ending at position 25 in s:

```
G G G G  G G A A------|
|  |  |  |  •  |  |  |       |
C C  C C  G  T  T T------|
```

Its loop length is 9, its number of errors is 1, and its score is 39.

Solution 3, beginning at position 3 and ending at position 25 in s:

```
G G G G A A A G ------|
|  |  |  |  •  |  |  |       |
C C  C C G T  T  T ------|
```

Its loop length is 7, its number of errors is 1, and its score is 39.

There are tiny position differences between these three solutions, and just one of them can be part of a secondary structure. This kind of situation is common, which explains the redundancy obtained by the Sagot-Viari algorithm. To solve this problem, we just have to keep, among a set of similar helices, the one that has the highest score. In the example above we choose the solution 1, as it has the highest score (41), and discard solutions 2 and 3.

To implement the procedure that calculates the scores, the parameter *minscore* mentioned-above is read from an input file Param, and a stack is used to keep track of all the helix scores. The algorithm is described in Figure 4.7.

```
Definition
StackScore(m)      set of helix scores corresponding to the model m

Other definitions and program are the same with those in Figure 3.4

Algorithm ManberSearch (m', ODI(m), StackScore(m), StackError(m),
                        PODI(m'), StackScore(m'), StackError(m'))
1:  while get the next elements w from ODI(m), s from StackScore(m) and e from
        StackError(m)
2:      extend w in sequence and get wβ
3:      if w is a direct occurrence
4:          if α = β  // where α is the last element of model m'
5:              put wβ in set PODI(m'), s in StackScore(m'), e in  StackError(m')
6:      else                                  // w is a complementary inverse occurrence
7:          if α match β (A-T or  G-C or G-T)
8:              s ← s+7 (for C-G) or  s ← s+5 (for A-T) or s ← s+3 (for G-T)
9:          else  if e+1 ≤ maxerror
10:                 s ← s –2,  e ← e+1
11:             else goto 1               // w is dropped
12:         put wβ in set PODI(m'), s in StackScore(m'), e in StackError(m')
13: end while

/*The sets PODI(m') StackScore(m') and StackError(m') are then the after
elongated elements.*/
```

Figure 4.7 Calculating scores for filtering helices.

4.3.2 Introducing the G-T base-pairing

In addition to Watson-Crick base-pairings, the non-canonical G-T pair is very frequently present in secondary structures. As reflected by the scores of Table 4.1, its thermodynamic stability is lower than that of Watson-Crick pairs, but higher than all other non-canonical pairs. The Sagot-Viari algorithm treats G-T pairs as substitution errors. Thus, if we want the program to output helices containing G-T pairs, the maximal allowed number of errors *errmax* should be large. For example, consider the lowing helix structure:

```
G U A C T G C A------ |
|  |  •  |  °  °  |  |       |
C A G G G T G U------ |
```

If we want to search for this kind of structure and if G-T is not considered as an error, then the maximum number of error allowed can be *errmax=1*. However, if G-T is considered as an error, then the helix contains three substitution errors, and to be able to find it we should set *errmax = 3*. The problem with increasing the maximal number of allowed errors is that the program finds a large set of irrelevant helices, that is, a large number of false positives. In order to be able to output helices containing G-T pairs without increasing the number of bad solutions, we consider the G-T pair as a valid base-pairing, as for A-T and G-C base-pairings. The only difference between the three base-pairings is in the way to evaluate them in a general helix score in the filtering process. The result is that now a model can have more than one complementary inverse. For example, the model TAG have the set of complementary inverse {CTA,TTA,CTG,TTG}. The algorithm dealing with G-T pairs is shown in Figure 4.7.

4.3.3 Conserved nucleotides

As mentioned in Chapter 2, a conserved base-pairing is a base-pairing (for example G-C) present in almost all the sequences of an alignment. An example is given by the helix III of the consensus structure (see Figure 4.4) corresponding to mitochondrial 5S rRNA (Figure 4.3). This helix contains two conserved base-pairings G-C and A-T situated just before the hairpin loop. If we search for helix element III without considering the conserved base-pairings, the program will output a large number of solutions that are not close enough to the consensus. Therefore, it is important to introduce in the program the constraints that characterize a given helix.

To be able to consider such kind of constraints in the program, we introduce, for each helix, an array *BaseCR* described in the input file Param. For each position *i* of the palindromic model *m* describing the helix, *BaseCR*[i] takes the value 0, 1, 2, 3 or –1 corresponding respectively to A, C, G, T or no conserved nucleotide. For example, consider the following description of a helix:

```
N N N A C N G G T------|
| | | | | | | | |     |
N N N T G N C C A------|
```

Its spreading out presentation may be:

N N N A C N G G T ------ A C C N G T N N N

N stands for any nucleotide (A, C, G or T). Thus, a pairing N-N means that it can be anything, that is, it is not a conserved base-pairing. The above helix contains 5 conserved base-pairings. The palindromic model describing this helix is *m*=ACCNGTNNN. The corresponding array *BaseCR* is:

BaseCR=[0, 1, 1, -1, 2, 3, -1,-1,-1]

Notice that we are able to define conserved nucleotides just in the lower side of a helix stem. Indeed, a conserved helix is defined through the palindromic model corresponding to the lower side of the helix. Thus, if only Watson-Crick pairings are allowed, then this palindromic model has only one possible complementary inverse, and the helix is uniquely defined. Otherwise, if the G-T pair is also allowed, then the palindromic model does not define the helix in a unique way.

Before describing the algorithm that takes the array *BaseCR* into account, we show how to consider a possible uncertainty in conserved base-pairings.

4.3.4 Introducing errors for primary structure constraints

As mentioned before, a conserved nucleotide is a nucleotide that is present at the same position in almost all the sequences of an alignment. This formulation means that, in some cases (in few sequences) this conserved nucleotide can be missing. Therefore, if we don't want the algorithm to be too restrictive, we should introduce a new parameter accounting for the uncertainty in the primary structure constraints. In other words, instead of having only one parameter *errmax*, we will have two parameters: *errmax* and *maxcbe*, the first one for the maximal number of allowed base-pairing errors, and the second for the maximal number of allowed conserved nucleotides errors.

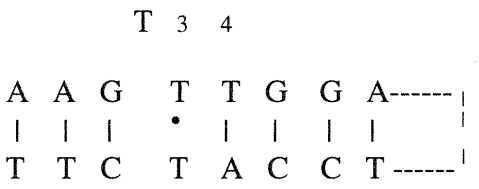


Figure 4.8 The P4 helix of Schizosaccharomyces pombe(X54421 – entry name in GenBank) mitochondrial Rnase P RNA. The fourth base-pairing is a valid T-A replacing a conserved C-G. The third base-pairing is absent and replaced by a non-canonical T•T.

For example, the structure of mitochondrial RNase P RNAs (Figure 4.5) contains a conserved helix P4, for which a possible consensus structure has been defined in Figure 4.6. This conserved structure contains four well-conserved base-pairings: A-T, G-C, C-G and C-G. However, in some cases, the fourth conserved base-pairing is absent. For example, in the case of the species Schizosaccharomyces pombe which P4 helix is shown in Figure 4.8, the C-G base-pairing is replaced by a T-A base-pairing. Observe that this T-A base-pairing is still a valid Watson-Crick base-pairing. Thus, it is not a secondary structure error, but a primary structure error. Sometimes both kinds of errors can be present in a helix, which is the case for Schizosaccharomyces pombe(Figure 4.8). Indeed, observe that the third conserved base-pairing is replaced by a non canonical T•T. This pairing accounts both for

secondary structure and primary structure errors, because it replaces the C-G pair of the bacterial consensus and also it has negative score.

Definition

BasseCR[k]	parameter array describing conserved nucleotide in each position of m
cbemodel	global variable used in counting the conserved base errors, initialized as 0
increase[i]	global Boolean variable array indicating whether or not there is a conserved nucleotide error in position i of m, initialized as false
maxcbe	maximum conserved base error allowed

Other definitions and program are the same with those in Figure 3.4 and Figure 4.7. The k^{th} iteration in constructing a model m'=mα is:

```
Algorithm ConstructModel ( seq, m, hmax, maxerror, ODI(m), k, PODI(m) )
1:  if ( k ≤ hmax )
2:    for α = A, C, G, T
3:      if BaseCR[k] ≠ α            //conserved base at position k of model ≠ α
4:        if cbemodel < maxcbe    //counted conserved base error <allowed
                                  // maximum value
5:          m' = mα             //  add α to the end of model
6:          cbemodel ← cbemodel + 1
7:          increase[k] ← true
8:        else go to 2
9:      else m' = mα
10:     ManberSearch ( m', ODI(m), StackScore(m), StackError(m), PODI(m'),
                       StackScore(m'), StackError(m') )    // See Figure 4.7
11:     Verify (POD(m'), POI(m'), OD(m'), OI(m') )    //See Figure 3.4
12:     ConstructModel (seq, m', hmax, maxerror, ODI(m'), k+1, PODI(m'))
13:     remove α from the end of the model
14:     if increase[k] = true
15:     cbemodel ← cbemodel - 1
16:     increase[k] ← false
17:   end for
18: end if
```

Figure 4.9 Dealing with the conserved nucleotides and their errors in constructing the model.

The algorithm for dealing with the conserved nucleotides is shown in Figure 4.9. To account for primary structures errors, we introduce two new parameters *maxcbe* and *cbemodel*, and an array *increase*. *maxcbe* is the maximal number of allowed

mismatches for the conserved nucleotides of a helix. It is read from the input file Param. *cbemodel* is used to count the conserved nucleotides errors in the process of building the helix model. It is set to 0 at the beginning and it cannot exceed *maxcbe*. *increase*[i] is a boolean parameter indicating whether or not the position i of the palindromic model has a conserved nucleotide error. At the beginning, we set *increase[i]* = *false* for every position i of the helix model. If a nucleotide other than the conserved one is set at position i when building the model, *increase*[i] is set to true, and *cbemodel* is increased by 1.

The model construction is performed by traversing a branching tree in the depth-first way. When the algorithm returns to a lower level i after finishing the construction of a model of size i+1, it is important to check *increase*[i]. If it has been set to true, that means that at level i *cbemodel* has been increased by 1, and thus we should decrease *cbemodel* by 1 and restore increase[i] to false.

4.3.5 Subsets of nucleotides

In Section 2.3, we mentioned that an alignment can be described through a consensus matrix, and that such a matrix can be used to define consensus nucleotides. However, if we want to define a conserved structure more precisely, we should extract more information from the consensus matrix, and represent each position by the subset of nucleotides that are most frequently present at that position. For example, we can ignore all nucleotides that have a frequency less than 0.05 at a given position.

For example, let us consider the consensus matrix of the $T\Psi C$ region of tRNAs (Table 2.2). We can notice that the most frequent nucleotides at position 48 are the pyrimidines Y={C,T}, and the most frequent nucleotides at position 57 are the purines R={A,G}. With the notations of Table 3.1, we can represent the $T\Psi C$ region by the consensus structure represented at the right of Figure 4.1. Similarly, the P4

helix of the mitochondrial RNase P RNA can be represented by the conserved structure of Figure 4.6.

In order to consider such nucleotides subsets in the definition of a helix in the input file Param, a two dimensional array *BaseCR*[k][j] is adopted instead of the ancient one dimensional array. The first index is the position in the helix model and can vary between 0 and the length of the helix stem. The second index is the number of nucleotides in the considered subset, and can take the values 0, 1 or 2, as there is a maximum of three nucleotides in each subset other than N. *BaseCR*[k][j] can take the values 0, 1, 2, 3 or –1 corresponding respectively to the nucleotide A, C, G, T or non-conserved nucleotide (that is the character N).

For example, consider the palindromic model: NNNLG. The program begins by initializing each value of *BaseCR*[k][j] to –1, for $0 \leq k \leq 4$ and $0 \leq j \leq 3$. When the program reads the model, it changes the parameter *BaseCR* depending on the characters encountered. For the first three 'N', there are no changes for the corresponding values of *BaseCR*. Now, for the 'L' at position 3, as 'L' represents the subset {A C T}, we set:

BaseCR[3][0] = 0 corresponds to 'A';

BaseCR[3][1] = 1 corresponds to 'C';

BaseCR[3][2] = 3 corresponds to 'T'.

And for the last 'G' we set

BaseCR[4][0] = 2 corresponds to 'G'.

The algorithm for constructing a palindromic model by taking nucleotide subsets into account is shown in Figure 4.10.

Definition
BaseCR[k][j] two dimension parameter array describing conserved nucleotide in each position k of m and in each position j of the subset

Other definitions and program are the same with those in Figure 3.4, Figure 4.7 and Figure 4.9

```
Algorithm ConstructModel ( seq, m, h_max, maxerror, ODI(m), k, PODI(m) )
1:  if ( k ≤ h_max )
2:    for α = A, C, G, T
3:      if BaseCR[k][0] ≠ -1            // the character at position k is not N
4:        if BaseCR[k][0] ≠ α and
            BaseCR[k][1] ≠ α and
            BaseCR[k][2] ≠ α
5:          if cbemodel < maxcbe   //counted conserved base error <allowed
                                   // maximum value
6:            m' = mα              // add α to the end of model
7:            cbemodel ← cbemodel + 1
8:            increase[k] ← true
9:        else go to 2
10:     else m' = mα
11:   else m' = mα
12:   ManberSearch ( m', ODI(m), StackScore(m), StackError(m), PODI(m'),
                      StackScore(m'), StackError(m') )    // See Figure 4.7
13:   Verify (POD(m'), POI(m'), OD(m'), OI(m') )    //See Figure 3.4
14:   ConstructModel (seq, m', h_max, maxerror, ODI(m'), k+1, PODI(m'))
15:   remove α from the end of the model
16:   if increase[k] = true
17:     cbemodel ← cbemodel - 1
18:     increase[k] ← false
19:   end for
20: end if
```

Figure 4.10 Dealing with the nucleotides subsets in constructing a model of size k.

4.3.6 Single stranded regions

So far, the only considered conserved nucleotides were part of conserved base-pairings, that is integrated in the folded part of a helix. However, in many cases, primary constraints are also located in the loops and single stranded regions. For example, the most conserved nucleotides in the *TΨC* region of tRNAs are situated in

the *TΨC* loop (see Figure 2.1). Another example is the helix P4 of mitochondrial RNase P RNA (see Figure 4.6), where five conserved nucleotides are located just before the beginning of the folded region.

To be able to take into account such conservation in single stranded regions, a new array *DeletionL*[k] is introduced, that indicates whether the nucleotide at position k in one side of the helix should be paired or not with a corresponding element in the other side of the helix. Since we put single stranded regions in the model, the side of the helix with single nucleotides corresponds to the direct occurrences of the model and the other shorter side of the helix corresponds to the complementary inverse of a model.

For each position i of the palindromic model m describing the helix, *DeletionL*[i] takes the value 0 or 1 corresponding to paired nucleotide or single nucleotide respectively, and this is performed initially when the program reads the palindromic model in file Param. When the program searches for elongated words in the sequence, *DeletionL*[k] for the extended position k should be checked. If *DeletionL*[k] = 1, the inverse occurrence is not extended, while the direct occurrence continues to be extended in the sequence. If *DeletionL*[k] = 0, both direct and inverse occurrences continue to be extended in the sequence.

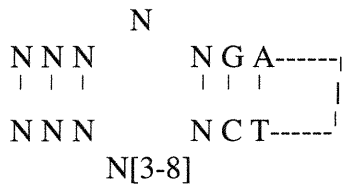
4.3.7 Conserved bulges and interior loops

Many conserved secondary structures contain bulges and interior loops at specific positions. For example, all mitochondrial 5S RNA structures that have been considered contain an interior loop situated in helix III (see figure 4.3, 4.4). It is always located at the same position, with one nucleotide at one side and three to eight nucleotides at the other side of the helix. Also, all the mitochondrial RNase P RNA structures that have been studied contain a bulge at a specific position in the helix P4 (see Figure 4.5, 4.6). It is formed by a unique nucleotide which is always a

T. The Sagot-Viari algorithm treats interior loops and bulges as errors (insertion and deletion). Therefore, it is not adapted to helices that have such loops as proper characteristics. Thus, a new method is introduced which can deal with conserved interior loops and bulges. The algorithm is given in Figure 4.11.

Four arrays are used: *BulminL*[k], *BulmaxL*[k] for the upper side of the helix, and *BulminR*[k], *BulmaxR*[k] for the lower side of the helix. The index k denotes the position in the palindromic model describing the helix. These four arrays indicate the minimal number and maximal number of nucleotides that can be inserted at the given position in the upper part and lower part of the helix respectively.

For example, consider the helix III of mitochondrial 5S RNA represented by the following conserved secondary structure:



N[3-8] in the lower part of the interior loop means that we can have a minimum of three nucleotides and a maximum of eight nucleotides. The four arrays *BulminL*, *BulmaxL*, *BulminL*, *BulminR* are set as follows. There is no conserved interior loop in the first part of the helix formed by the first three base-pairings N-N. Thus, for $0 \leq i \leq 2$ we set:

BulminL[i] = BulmaxL[i] = BulminR[i] = BulmaxR[i] = 0

The conserved internal loop begins at position 3, so we set:

BulminL[3] = 1, BulmaxL[3] = 1, BulminR[3] = 3, BulmaxR[3] = 8

Definition

BulminR[k]	constant arrays indicating the minimal number of nucleotides in the loop at the position k of the lower part of the helix
BulmaxR[k]	constant arrays indicating the maximum number of nucleotides in the loop at the position k of the lower part of the helix
BulminL[k]	constant arrays indicating the minimal number of nucleotides in the loop at the position k of the upper part of the helix
BulmaxL[k]	constant arrays indicating the maximum number of nucleotides in the loop at the position k of the upper part of the helix

Other definitions and program are the same with those in Figure 3.4 and Figure 4.7. Here we omit the details of treatment for score and helix error since they are the same with those in Figure 4.7.

Algorithm **ManberSearch'** (m', ODI(m), StackScore(m), StackError(m), PODI(m'), StackScore(m'), StackError(m'))

```
1:  while read the next elements w from ODI(m)
2:      if w is a direct occurrence    // the lower part of the helix
3:          if BulminR[k] = 0 and BulmaxR[k] = 0   //no loop or bugle
4:              extend w in sequence and get wβ
5:              if α = β            // where α is the last element of model m'
6:                  put wβ in set PODI(m'=mα)
7:          else for  i = BulminR, until BulmaxR          //there is a loop or bulge
8:                  extend w with i+1 elements in sequence, and get wb1b2...biβ
9:                  if  α match β
10:                     put wb1b2...biβ in set PODI(m'=mα)
11:             end for
12:     else                        //w is a inverse occurrence, in upper part of the helix
13:         if BulminL[k] = 0 and BulmaxL[k] = 0   //no loop or bulge
14:             extend w in sequence and get wβ
15:             if α match β
16:                 put wβ in set PODI(m'=mα)
17:             else  deal with the errors
18:         else for  i = BulminL, until BulmaxL          //there is a loop or bugle
19:                 extend w  with i+1 elements, and get wb1b2...biβ
20:                 if  α match β
21:                     put wb1b2...biβ in set PODI(m'=mα)
22:                 else  deal with the errors
23:             end for
24: end while
```

The set PODI(m'=mα) are then having elongated direct and inverse occurrences.

Figure 4.11 Dealing with conserved interior loops and bulges when searching for the elongated words.

For the example above, when the program searches for the upper side of the helix, at position 3 it jumps one position in the sequence, then it continues to search for the next position 4. When the program searches for the lower occurrence, at position 3 it jumps 3 elements, 4 elements, ..., until 8 elements, because it should search for all the possible cases: the size of the lower side of the loop can vary between 3 and 8. Then it continues to search for the next position 4.

This method is general. It can deal with the helix with several interior loops and bulges at specific positions, and the number of nucleotides in the loop or bulge can be variable just as explained in the above example.

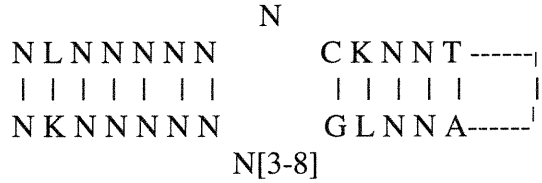
4.3.8 Representing the helices in the input file

Now that we have adapted the Sagot-Viari algorithm to various kinds of helices containing various primary and secondary structure constraints, we should describe a useful way to represent such helices in the input file. The new parameters that have been introduced in our algorithms (*BaseCR*[k][j], *BulminR*[k], *BulmaxR*[k], *BulminL*[k] and *BulmaxL*[k]) should be deduced from a specific representation of a helix.

A helix will be described by its lower side. The upper side is deduced from the lower side. Here is an example:

Motif = A N[2] L G#[1,1][3,8] N[5] K N[1]

This motif is a description of the following helix:



More precisely, a motif is a sequence of strings, where each string S can be of one of the following form:

- S is a character from the alphabet $\sum$={A,C,G,J,K,L,M,N,Q,R,S,T,W,Y,Z}. The meaning of each character is given in Table 3.1.
- S is a character X from $\sum$ followed by [i] where i is a given integer. This means that the corresponding helix contains X repeated i-times in tandem.
- S is a character C of $\sum$ followed by #[i,j][k,l], where i,j,k,l are four integers such that $i \leq j$ and $k \leq l$. This means that the helix contains the character C followed by an interior loop which upper side size varies between i and j, and lower side size varies between k and l. If one of the two intervals is [0], then the interior loop is a bulge.
- S is a character C of Σ followed by +. This means that only the lower side of the helix contains this character C in the corresponding position, while the up side of the helix contains a vacancy in this position.

Let us consider the model given above. Before reading the model, all the parameters are initialized: *BaseCR*[i] = -1, *BulminL*[i] = 0, *BulmaxL*[i] = 0, *BulminR*[i] = 0 and *BulmaxR*[i] = 0. When the program reads the first element of the motif, that is, the conserved nucleotide A, the parameter *BaseCR*[0][0] is set to 0; the second string N[2] means that there are two nucleotides at position 1 and 2. The third string is the character L corresponding to the subset {A, C, T}, so *BaseCR*[3][0] =0, *BaseCR*[3][1]=1 and *BaseCR*[3][2]=3. The fourth string is G#[1,1][3,8] meaning that the helix contains a conserved nucleotide G fallowed by an interior loop with the length constraints defined by the two given intervals. Therefore, *BaseCR*[4][0]=2, *BulminL*[4]=1, *BulmaxL*[4]=1, *BulminR*[4]=3 and *BulmaxR*[4]=8. The fifth string, N[5], means that the helix contains five consecutive nucleotides at positions 5 to 10, that can be any nucleotide from the alphabet {A,C,G,T}. The sixth string is the character K corresponding to the subset {A, G, T}, and thus *BaseCR*[11][0]=0,

BaseCR[11][1]=2 and *BaseCR*[11][3]=3. The last string corresponds to one last nucleotide.

A last notation (+) is introduced in our helix models to be able to consider conserved nucleotides in the single stranded regions. A string formed by a character of Σ followed by + will be considered as unpaired, that is, without a corresponding complement in the upper side of the helix. An example of such model is given. For example, the stem of the P4 helix of RNase P RNAs can be represented by:

Model = N[3] L C#[0,0][1,1] G A A A+ G+ G+ A+ K+

Before reading the model, *DeletionL*[i] is set to 0, for i = 0, 1, ... 12. When the first A+ is read, the program set *BaseCR*[8][0]=0 and *DeletionL*[8] =1. Similarly, *DeletionL*[k] = 1 for k = 9, 10, 11, 12.

4.4 Conclusion

Based on the Sagot-Viari algorithm, we have developed a flexible and efficient method that is able to identify, in a genomic sequence *G,* all the occurrences of a specific secondary structure. These secondary structures are helices defined by their stem and loop length, the presence of conserved nucleotides, the presence of conserved base subsets and the presence of interior loops and bulges at specific places. Our new algorithm allows for a very flexible representation and identification of different kinds of helices. Moreover our program is able to deal with the uncertainty in the primary structure, as well as in the secondary structure. In addition, introducing score calculation makes it possible for us to filter the solutions to avoid the redundancy in the output. Our program takes less spaces and times than that of Sago-Viari's algorithm, because when model is been constructing we consider just conserved nucleotides and conserved base subsets at some positions

instead of taking every one successively from the alphabet {A C G T} for every position of the model.

Chapter 5

Applications and results

In this chapter, we test our algorithm on specific annotated genomes. We search for all possible occurrences of 5S RNAs and RNase P RNA, and we compare the obtained results with the results reported in the literature. We also test the effect of different sets of parameters on the output.

To analyze the results of an algorithm, two types of errors should be considered:

- **The false negatives**: The sequences corresponding to "true" genes detected by other methods and reported in the annotations, but not found by our algorithm.
- **The false positives**: The sequences found by our algorithm, but that do not correspond to real genes.

As we restrict ourselves to the search of specific subsequences of a complex structure, the false positives can be explained by the fact that we did not consider all the constraints of a gene structure, and these false positives can be eliminated by the search of the other substructures. Therefore, obtaining false positives is not necessarily a bad result, in contrast with obtaining false negatives. Moreover, it is

usually better to obtain too many results than to miss possible occurrences, as false positives can then be tested by other ways.

5.1 Searching for mitochondrial 5S RNA

Mitochondrial 5S rRNAs are involved in mRNA translation (protein synthesis). The general characteristics of a 5S RNA secondary structure have been defined in Section 4.2.2. It is formed of four helices, among which helix III is the most constrained one. Each helix is defined by general constraints such as conserved base-pairings, a conserved interior loop for helix III, and specific length constraints. In order to define these constraints more precisely, we considered the structures of 10 specific mitochondrial 5S RNAs: the plants chondrus crispus (Z47547 – entry name in GenBank), marchantia polymorpha (X04465), triticum aedtivum, the algae cianidium caldarum, cyanidioschyzon merolae (D89861), plocamiocolax pulvinata, nephroselmis olivacea (AF137379) and prototheca wickerhamii (U02970), and the bacteria jakoba libera and reclinomonas Americana NZ. All of them have been well defined by other methods (for example, by x ray infraction), and we extract some general characteristics.

5.1.1 Choosing appropriate parameters

Table 5.1 summarizes the parameters used to define the two helices I and III. We did not take into account the other helices of the structure as they are very poorly conserved. *hmin* is the minimum helix length, *hmax* the maximum helix length, *lmin* the minimum loop length, *lmax* the maximum loop length, *errmax* the maximum number of errors allowed (base-pairings that are not Watson-Crick pairs), *indel* is the insertion/deletion permission (*indel=1* means insertion/deletion are allowed) and *minscore* is the minimum score used to filter the helices. *cberr* is the maximum number of errors allowed for conserved bases.

Parameter	element I	element III
hmin	6	6
hmax	10	6
lmin	96	13
lmax	103	13
errmax	1	0
indel	0	0
minscore	32	32
cberr	0	0

Table 5.1 The parameters used to define the elements I and III.

In addition to these constraints, helix element III is defined by a certain number of conserved base-pairings and an internal loop. These constraints are represented by the following model:

Model for helix III: T C N*[1,1][3,8] N[3]

5.1.2 Searching for 5S RNAs

The general method used to find all 5S RNAs in a genome is the following. First, we search for the helix elements I and III. Then we combine the different occurrences of these two elements by respecting the distance constraints: there is a minimum of 17 nucleotides and a maximum of 21 nucleotides between the helices I and III of a 5S rRNA structure.

```
--------
SEQUENCE R.amer mtDNA
--------

HELICE 3 lmin=13, lmax=13, scoremin=32, errmax=0, boucle: 1 et 3-8
--------

MODELS OF LENGTH 6
----------------

PALINDROME
 Pos1                    Score   Pos2
>67259  C A C T T G A   34      67279  T C G A A T T G T G
 56034  T G T T G G A   32      56054  T C T C A G A A T G G A C A
 47428  A T T T T G A   32      47448  T C A T A T A T A A T
 41187  A G T A T G A   32      41207  T C A T T A A G G C T
 32263  T T T T G G A   32      32283  T C C G A T A G A
 31439  T A T T A G A   32      31459  T C T T A T A G A T A
 31001  T T T T A G A   32      31021  T C T T T A A C T G A A A A
 30926  T T G T T G A   34      30946  T C A A A G C A A
 19887  C A G C A G A   32      19907  T C T T C G A T A A T T G
 13352  T T T T A G A   32      13372  T C T A A T T T T A A A
 6371   T A T T A G A   32      6391   T C T T T A A A T A
 1220   C T G C G G A   34      1240   T C T T G G T A G C A G
 343    T A T T A G A   32      363    T C T T G C C A T A
```

Figure 5.1 The output of our program for searching all occurrences of helix III in the R_amer_mtDNA sequence of the Reclinomonas Americana genome. The first occurrence marked with symbol '>' is the only one that is part of a real 5S rRNA structure, and it is the one given in Figure 4.3.

To give an example let us consider the sequence R_amer_mtDNA of Reclinomonas Americana. This sequence is of size 114, and the only 5S rRNA structure reported in the annotation of that sequence is the one given in Figure 4.3. We detail here the results obtained by our algorithm for searching 5S rRNA sequences in this genomic fragment:

- Two thousand occurrences of element I have been obtained. This is because element I has not a very constrained structure: it does not contain conserved base-pairings, and its stem is not very long.
- 13 occurrences of element III have been found. Figure 5.1 is the output of our program for searching occurrences of element III.
- After combining the occurrences of element I with the occurrences of element III, two solutions remain. One of them corresponds to the unique 5S rRNA reported in the annotations of the R_amer_mtDNA sequence of Reclinomonas Americana (see Figure 4.3). In Figure 5.1, the first one (at position 67259) is the occurrence of helix III that is a part of this final structure.

5.1.3 The effect of filtering

As discussed in Section 4.3.1 of Chapter 4, in order to avoid the redundancy in the solutions found by the Sagot-Viari algorithm, we introduced a score calculation and filtered the solutions by choosing the one that had the best score. This had the effect of considerably reducing the number of output results. For example, we have searched all the occurrences of helix I in the sequence of Nephroselmis olivacea. The effect of this filtering can be seen when we compare the results obtained with and without filtering. We can see in Figure 5.2 that filtering largely cuts down the number of occurrences obtained, especially for helix length 8 (29.64% reduction) and 9 (34.70% reduction).

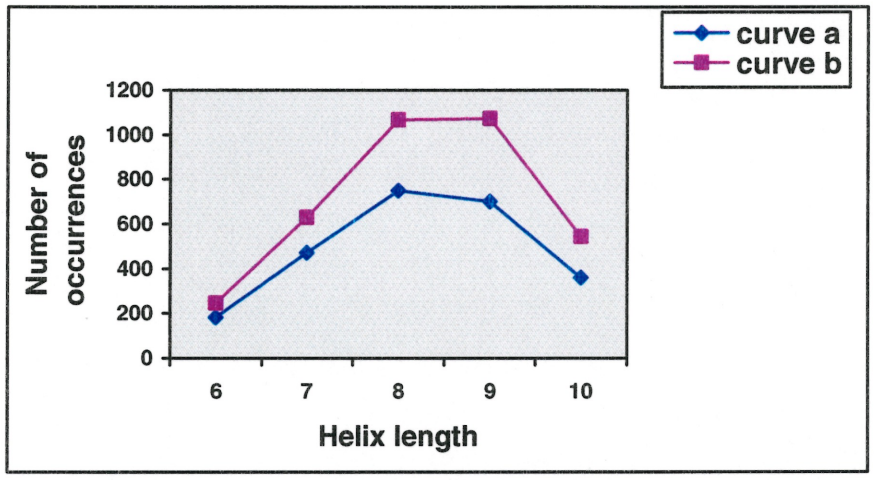


Figure 5.2 The number of occurrences of helix element I vs. the helix length in the sequence of Nephroselmis olivacea. The helix length vary between *hmin*=6 and *hmax*=10, as given in Table 5.1. Curve a is the results obtained when no filtering is considered, and curve b is the results obtained with our new filtering.

5.1.4 Results

We have searched for all potential sequences of 5S rRNAs in four genomic sequences: Chondrus crispus(size: 25836; entry: Z47547), Nephroselmis olivasea (size: 200799; entry: AF137379), Prototheca wickerhamii(size: 55328; entry: U02970) and Reclinomonas Americana(size: 69034; entry: AF007261). Table 5.2 presents the number of solutions found out by our method in the four genomic sequences. We can see that there are no false negatives in our solutions, yet several false positive may exist for some sequences.

Sequence	Solution	False positive	False negative
C. crispus	1	0	0
N. olivasea	3	2	0
P. wick.	4	3	0
R. Americana	2	1	0

Table 5.2 The number of helices found out by combining the occurrences of element I with the occurrences of element III with respecting the distance constraint. The number of false positive and the number of false negative for the four sequences are listed as well.

Figure 5.3 shows the number of occurrences obtained for helix I depending on its length, for each of the four genomic sequences. As we have noticed, helix element III is much more constrained than helix element I, and thus, far less occurrences should be found for this helix. This is demonstrated by comparing the number of occurrences of helix I (Figure 5.2) and the number of occurrences of helix III in Table 5.3. This table shows the number of occurrences of helix element III of length 6 in the four genomic sequences, with the two specific characteristics of helix III (interior loop and conserved base-pairings) are considered, and when only length constraints (parameters of Table 5.1.) are considered. One can see from this table that the number of occurrences have been enormously reduced by taking all characteristics of helix III into account.

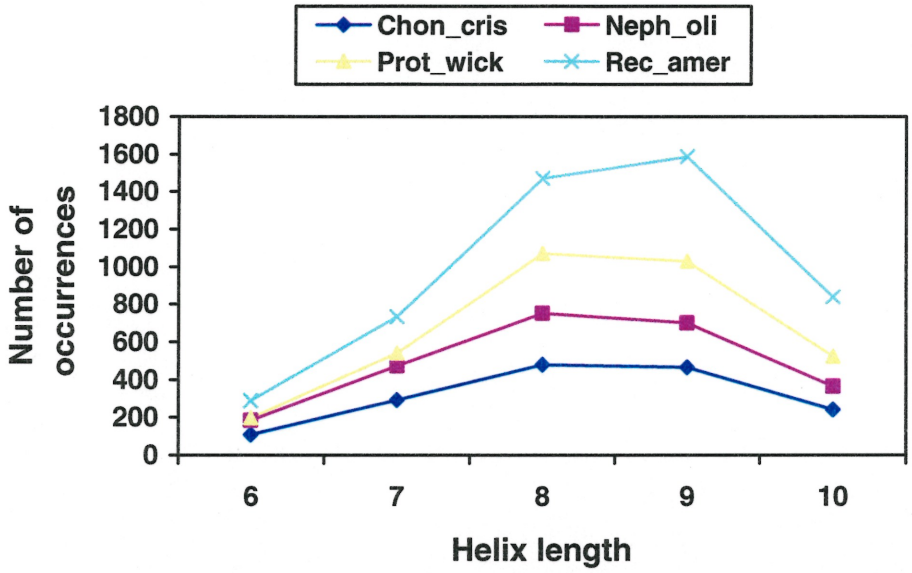


Figure 5.3 The number of occurrences vs. the model length for helix element I. The curves correspond to the sequences: chondrus crispus mt DNA, nephroselmis olivacea mt DNA, prototheca wickerhamii mt DNA and reclinomonas americana mt DNA.

Sequence	Number of occurrences (length 6)	
	No improvement	With two improvements
Chon-crise	128	3
Neph-oli	222	8
Pro-wick	235	11
R-amer	326	13

Table 5.3 The number of occurrences for helix element III in four sequences, with two improvements (interior loop and conserved base pairs) and without any improvement (only considering the length constraint).

5.2 Searching for mitochondrial RNase P RNA

The general characteristics of a mitochondrial RNase P RNA secondary structure have been given in Section 4.2.3. It contains a certain number of conserved substructures, the most constrained one being helix element P4 (see Figure 4.5, 4.6, 4.7).

We first focus on the identification of all occurrences of helix P4. We then verify, with the surrounding helix P1 whether the whole sequence is a potential RNase P RNA.

5.2.1 Choosing appropriate parameters

We have considered mitochondrial RNase P RNAs of 10 bacterial: Jakoba libera, Reclinomonase Americana NZ, Schzosaccharomyces pombe, Saccharomyces cerevisiae(AJ011856), Schizosaccharomyces octosporus(X54421), Saccharomyces exiguous(AJ011856), nephroselmis olivacea, Rhizopus stolonifer, Emericella nidulans and mucor mucedo. Table 5.4 summarizes the parameters used to define the helices P1 and P4.

In addition, two models are defined for helices P1 and P4:

Model for helix P1: N*[0,2][0,0] N[3] N*[0,0][0,1] N*[0,1][0,0] N[8];

Model for helix P4: N[3] L L*[0,0][1,1] G N[1] A A+ G+ G+ L+ K+.

Parameter	Helix P1	Helix P4
hmin	11	13
hmax	16	13
lmin	100	80
lmax	900	820
errmax	0	2
indel	1	0
minscore	53	37
cberr	0	0

Table 5.4 The parameters used to define the helices P1 and P4.

5.2.2 Searching for RNase P RNAs

First, we search for the helix elements P1 and P4. Then, we combine the different occurrences of these two elements by respecting the distance constraints: there is a minimum of 1 nucleotide and a maximum of 2 nucleotides between the elements P1 and P4 (shorter side) of a RNase P RNA structure (see Figure 4.5, 4.6).

To give an example, let us consider the sequence of mitochondrial RNase P RNA of Reclinomonas Americana NZ. This sequence is of size 315 and the only RNase P RNA structure reported in the annotation of that sequence is the one given in Figure 4.5. We detail here the steps of searching for RNase P RNA sequences by our algorithms in this genomic fragment:

- 20 000 occurrences of helix P1 have been found.
- 10 occurrences of helix P4 have been found. Figure 5.4 is the output of our program for searching occurrences of helix element P4.

- After combining the occurrences of helix element P1 with the occurrences of helix element P4, two solutions remain. One of them corresponds to the unique RNase P RNA reported in the annotations of the R_amer_mtDNA sequence of Reclinomonas Americana NZ (see Figure 4.5). In Figure 5.4, the one marked with '>' (at position 33500) is the occurrence of helix P4 that is a part of this final structure.

```
--------
SEQUENCE R.amer mtDNA reversed
--------

MODELS OF LENGTH 13
----------------

PALINDROME
 Pos1                    Score Pos2
 66525  T T T G G G T C  39    67046   G C C C C A G A A A G G C A
 61584  T T T T T A A A  38    61726   T T T A A A G A A A G G T T
 61354  T T C T T A A A  42    61726   T T T A A A G A A A G G T T
 60189  T A T A A A A A  38    60928   T T T T T C G T A A G G A T
 50124  T G C T A T G A  38    50475   T T A T A T G T A A G G A A
 44856  T A T T G G T G  37    45249   C A C C A G G A A A G G T T
>33500  T T C G A C C T  48    33751   A G G T C T G A A A G G A T
 33173  T C T G A C C T  37    33751   A G G T C T G A A A G G A T
 26435  T T C T A A A A  42    26550   T T T T A T G A A A G G T G
 13474  T T T G G A T G  42    14161   C A T C C T G G A A G G T A
```

Figure 5.4 The output of our program for searching all occurrences of helix P4 in the RNase P RNA sequence of the Reclinomonas Americana genome. The occurrence marked with symbol '>' is the only one that is part of a real RNase P RNA structure, and it is the one given in Figure 4.5.

5.2.3 Results

We have searched for RNase P RNA sequences in the four DNA sequences corresponding to the four bacteria: Rhizopus stolonifer(size: 54178), Reclinomonas

Americana NZ (size: 69034; entry: AF007261), Nephroselmis olivacea (size: 200799; entry: AF137379) and Schizosaccharomyces pombe (size: 19431; entry: X54421). Table 5.5 summarizes the solutions found. We can see that there are no false negative in our solutions, yet one false positive exist for some sequences.

Sequence	Solution	False positive	False negative
R. stolonifer	1	0	0
N. olivasea	2	1	0
S. pombe	2	1	0
R. Americana	2	1	0

Table 5.5 The number of helices found out by combining the occurrences of element P1 with the occurrences of element P4. The number of false positive and false negative is given.

5.2.4 Discussion on parameter settings

To give a better idea about the influence of the parameters in the searching algorithm, we compare the results obtained for searching the helix P4 in the sequence of the mitochondrial genome of S.pombe, by considering different parameter sets. Table 5.6 gives five parameter sets and the corresponding number of occurrences found in the genome. The number of special characters refers to any character except A, C, G, T, N in the model defining the helix (characters defining a set of nucleotides). In this table, we just report the set of parameters that leads to no false negative.

Set	Number of occurrences	Number of special characters	Conserved base error	Base pair error
Set no.1	237	0	4	2
Set no.2	134	1	3	2
Set no.3	63	2	2	2
Set no. 4	24	3	1	2
Set no. 5	5	4	0	2

Table 5.6 The parameter sets for searching the structure P4, and the corresponding number of objects found in S. pombe mt sequence.

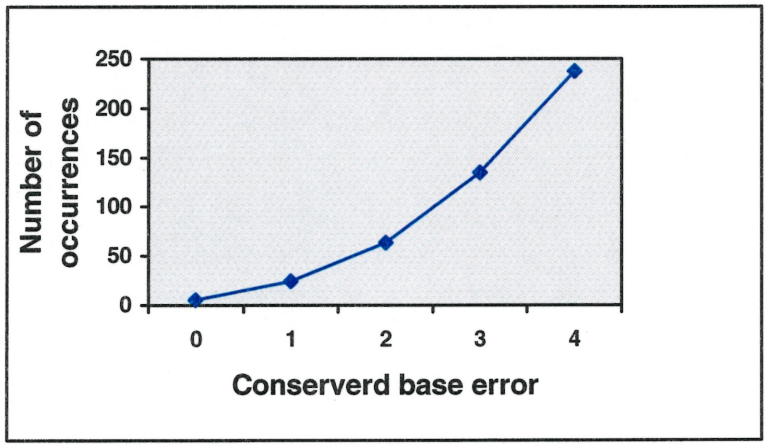


Figure 5.5 The number of helices found greatly increases as the conserved base error increases from 1 to 4, though the number of base subset in the model decreases.

Although helix element P4 has been found out with each of these five sets, the difference in the number of false positives is large (Figure 5.5). The number of occurrences greatly increases as the number of allowed errors on the conserved nucleotides increases from 0 to 4, though the number of special characters in the model decreases. We have performed the same kind of experiences on the four genomes: Rhizopus stolonifer, Reclinomonase Americana NZ, Nephroselmis olivacea and Schzosaccharomyces pombe. Table 5.7 is the result of searching P4 by choosing a model that does not contain any special character, and by varying the number of errors allowed. Conversely, Table 5.8 is the result of searching P4 by allowing no error on the conserved bases, and by varying the number of special characters in the model.

Conserved base error	False positive	False negative
0	2	3
1	6	3
2	57	1
3	413	1
4	2320	0

Table 5.7 Searching P4 with a model that does not contain any special character. As the number of conserved base error increases, the number of false positive increases, and the number of false negatives decreases. Here the maximum base pair error is 2.

Number of base subset	False positive	False negative
0	2	3
1	5	3
2	8	1
3	14	1
4	34	0

Table 5.8 Searching P4 by varying the number of special characters in the model. Here the maximum base pair error is 2.

These results show that the effect of errors is higher than that of special characters. This is because a special character influences just one position, whereas an increase in the number of errors allowed influences every position. So we conclude that the number of occurrences slightly increases with the number of special characters in the model, but highly increases with the number of allowed errors.

Chapter 6

Conclusion

Based on the Sagot-Viari algorithm, we have developed a flexible and efficient method for identifying, in a genomic sequence *G*, all the occurrences of some specific secondary structure. These secondary structures are helices defined by their stem and loop length, the presence of conserved nucleotides, the presence of conserved base subsets and the presence of interior loops and bulges at specific places. Such an algorithm for searching helices can be used in the different existing methods of complex biological structure search (RNAmot, RNAbob, Palingol) to improve the flexibility of these algorithms, or can be the basis of another general method. The idea is to first subdivide the general structure into a set of helices, to search for all these helices in the genome being analyzed by our algorithm, and then to assemble the different substructures. In that way, the user can try to assemble the "basic" helices in different ways, and new structures can be more easily discovered.

Our new algorithm makes it possible to easily represent different kinds of helices. Besides, based on the score calculation, we improved the output of the algorithm by filtering the solutions and selecting the best results between all possible occurrences. These occurrences are chosen on the base of a general score. The filtering had the effect of considerably reducing the number of output results.

We used our algorithm for searching different structures in various genomes. In each case, the algorithm identifies the annotated RNAs, with no false negatives and a few number of false positives. Moreover, by testing the effect of different parameters sets on the output, we noticed that the number of occurrences slightly increases with the number of special characters in the model, but highly increases with the number of allowed conserved base errors.

References

[AGM90] Altschul, S.F., Gish, W., Miller, W., Myers, E.W. and Lipman, D.J. «Basic local alignment search tool». *Journal of Molecular Biology,* 215:403-410, 1990.

[BKV96] B. Billoud, M. Kontic, and A. Viari. «Palingol: a declarative programming language to describe nucleic acids' secondary structures and to scan sequence databases». *Nucleic Acids Research*, 24(8): 1395-1403, 1996.

[Cec93] Cech T.R. «Structures and mechanism of the large catalytic RNAs: Group I and group II introns and ribonuclease P». In Gesteland, R.F. and Atkins, J.F. (eds), The RNA World. *Cold Spring Harbor Press*, New-York, pp. 239-270, 1993.

[dCBT90] Y. d'Aubenton Carafa, E. Brody, and C. Thermes. «Prediction of rho-independant E.coli transcription terminators. *Journal of Molecular Biology*, 216: 835-858, 1990.

[ED94] S R. Eddy and R. Durbin. «RNA sequence analysis using covariance models». *Nucleic Acids Research*, 22(11): 2079-2088, 1994.

[ECC76] B. Ecarot-Charrier and R. J. Cedergren. «The preliminary sequence of $tRNA_F^{Met}$ from anacystis nidulans compared with other initiator tRNAs». *Federation of European Biological Sciences Letters* 63: 287-290, (1976)

[Edd96] S.R. Eddy. «rnabob – search for RNA motifs in sequence databases». 1996. Manuscript.

[EML96] N. El-Mabrouk and F. Lisacek. «Very fast identification of RNA motifs in genomic DNA. Application to tRNA search in the yeast genome». *Journal of Molecular Biology*, 264: 46-55, 1996

[FB91] G A. Fichant and F. Burks. «Identifying potential tRNA genes in genomic DNA sequences». *Journal of Molecular Biology*, 220: 659-671, 1991.

[FW75] G.E. Fox and C.C. Woese. «5S RNA secondary structure». *Nature*, 256, 505-507, 1975.

[GGM98] P. Gendron, D. Gautheret and F. Major. «Structural ribonucleic acid motifs identification and classification». In *High Performance Computing Systems and Applications, Kluwer Academic Press*, chapter 31, 1998.

[GMC90] D. Gautheret, F. Major, and R. Cedergren. «Pattern searching/alignment with RNA primary and secondary structures». *Computer Application in Bioscience*, 6(4): 325-331, 1990.

[JGS84] A.B. Jacobson, L. Good, J. Simonetti and M. Zuker. «Some simple computational methods to improve the folding of large RNAs». *Nucleic Acids Research*, 12, 45-52,1984.

[JOP89] B.D. James, G.J. Olsen and N.R. Pace. «Phylogenetic comparative analysis of RNA secondary structure». *Methods in Enzymology*, 180, 227-230, 1989.

[KKB94] V. Knoop, S. Kloska, and A. Brennicke. «The identification of group ii introns in nucleotide sequence data». *Journal of Molecular Biology*, 242: 389-396, 1994.

[Kon93] M. Kontic. «Palingol. Langage pour la description et la recherché de structures secondaires dans les sequences nucléotidiques». *DEA d'Intelligence Artificielle*, Université de Paris Nord, 1993.

[LBK97] B.F. Lang, G. Burger, C.J. O'Kelly, R.J. Cedergren, B. Golding, C. Lemieux, D. Sankoff, M. Turmel, and M.W. Gray. «An ancestral mitochondrial DNA resembling a eubacterial genome in miniature». *Nature,* 387: 493-497, 1997.

[LDM94] F. Lisacek, Y. Diaz, and F. Michel. «Automatic identification of group I introns cores in genomic DNA sequences». *Journal of Molecular Biology*, 235: 1206-1217, 1994.

[LE97] T.M. Lowe and S.R. Eddy. «tRNAscan-SE: a program for improved transfer RNA detection in genomic sequence». *Nucleic Acids Research* 25, 955-964, 1997.

[LE99] T.M. Lowe and S.R. Eddy. «A computational screen for methylation guide snornas in yeast». *Science*, 283(5405): 1168-1171, 1999.

[Mar83] H.M. Martinez. «An efficient method for finding repeats in molecular sequences». *Nucleic Acids Research*, 11:4629-4634, 1983.

[MJW98] C. Massire, L. Jaeger, and E. Westhof. «Derivation of the three-dimensional architecture of bacterial ribonuclease P RNAs from comparative sequence analysis». *Journal of Molecular Biology*, 279: 773-793, 1998.

[MM89] E.W. Myers and W. Miller. «Approximate matching of regular expressions». *Bulletin of Mathematical Biology*, 51(1):5-37, 1989.

[MUO89] F. Michel, K. Umesono, and H. Ozeki. «Comparative and functional anatomy of group ii catalytic introns – a review». *Gene*, 82: 5-30, 1989.

[Mye96] E.W. Myers. «Approximate matching of network expression with spacers». *Journal of Computational Biology*, 3(1):33-51, 1996.

[NJ80] R. Nussinov and A. Jacobson. «Fast algorithm for predicting the secondary structure of single stranded RNA». *Proceedings of the National Academy of Sciences, USA*, 77, 6309-6313, 1980.

[OB00] R. Overbeek and S. Brunetta. «PatScan», internet addresses: http://bio-www.ba.cnr.it:8000/BioWWW/patscanhelp.html.

[OLR00] A.D. Omer, T.M. Lowe, A.G. Russell, H. Ebhardt, S.R. Eddy, and P.P. Dennis. «Homologs of small nucleolar rnas in archaea». *Science*, 288(5465): 517-522, 2000.

[PCB94] A. Pavesi, F. Conterio, A. Bolchi, G. Dieci, and S. Ottonello. «Identification of new eukaryotic tRNA genes in genomic DNA databases by a multistep weight matrix analysis of transcriptional control regions». *Nucleic Acids Research*, 22(7): 1247-1256, 1994.

[PL88] W R. Pearson and D J. Lipman. «Improved tools for biological sequence comparison». *Proceedings of the National Academy of Sciences*, USA **85**; 2444-2448, April 1988.

[RRB76] A. Rich and U.L. Raj-Bhandary. «Transfer RNA: molecular structure, sequence, and properties». *Annual Review of Biochemistry*, 45:805-860, 1976.

[SBM94] Y. Sakakibara, M. Brown, I.S. Mian, R. Underwood and D. Haussler. «Stochastic context-free grammars for modeling RNA». In *Proceedings of the Hawaii International Conference on System Sciences.* IEEE Computer Society Press, los Alamitos, CA, 1994.

[Sea93] D.B. Searls. «The computational linguistics of biological sequences». In *Artificial Intelligence and Molecular Biology.* AAAI Press, Menlo park, CA, pp. 47-120., 1993.

[SK83] D. Sankoff and J. B. Kruskal, (ed.) «Time warps, string edits and macromolecules: the theory and practice of sequence comparison», Addison Wesley, Chapter three, 1983.

[Sta80] R. Staden. «A computer program to search for tRNA genes». *Nucleic Acids Research*, 8(4):817-825, 1980.

[SV97] M.F. Sagot and A. Viari. «Flexible identification of structural objects in nucleic acid sequences: palindromes, mirror repeats, pseudoknots and triple helices» In A. Apostolico and J. Hein, editors, *LNCS*, volume 1264 of *CPM97*, pages 224-246. Springer, 1997.

[TUL71] L.J. Tinoco, O.C. Uhlenbeck and M.D. Levine. «Estimation of secondary structure in ribonucleic acids». *Nature*, 230, 363-367, 1971.

[Tur88] D.H. Turner. «RNA structure prediction». *Annual Review of Physical Chemistry*, 17:167-192, 1988.

[Wat89] M.S. Waterman. «Consensus methods for folding single-stranded nucleic acids». In M.S. Waterman, editor, *Mathematical Methods for DNA Sequences,* pages 185-224. CRC Press, 1989.

[WGG83] C.R. Woese, R.R. Gutell, R. Gupta and H.F. Noller. «Detailed analysis of the higher-order structure of 16S-like ribosomal ribonucleic acids». *Microbiology Review*, 47, 621-669, 1983.

[ZS81] M. Zuker and P. Stiegler. «Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information». *Nucleic Acids Research*, 9, 133-148, 1981.