Searching for alternate RNA structures in genomic sequences

Azadeh Saffarian, Mathieu Giraud, Hélène Touzet

Searching for alternate RNA structures in genomic sequences

Azadeh Saffarian, Mathieu Giraud, and Hélène Touzet

LIFL (UMR 8022 CNRS, University of Lille) and Inria Lille, France

Abstract. We introduce the concept of *RNA multi-structures*, that is a formal grammar based framework specifically designed to model a set of alternate RNA secondary structures. Such alternate structures can either be a set of suboptimal foldings, or distinct stable folding states, or variants within an RNA family. We provide several such examples and propose an efficient algorithm to search for RNA multi-structures within a genomic sequence.

1 Introduction

Structural RNAs play a wide range of roles in the cell of all organisms. In many RNA families the spatial architecture of the molecule is an important component of its function [1, 9]. This spatial architecture is mainly built upon the set of base pairings, that is captured in the secondary structure. Over the years, a great number of computational methods have been proposed to model consensus secondary structures. In many cases however, the signature for an RNA family cannot be compiled into a single consensus structure, and is mainly given by *a set of alternate secondary structures*. For example, certain classes of RNAs adopt at least two distinct stable folding states to carry out their function. This is the case of riboswitches, that undergo structural changes upon binding with other molecules [5], and recently some other RNA regulators were proven to show evolutionary evidence for alternative structure [11]. The necessity to take into account multiple structures also arises when modeling an RNA family with some structural variation across species, or when it comes to work with a set of predicted suboptimal foldings. The objective of this paper is to propose a model to deal with such set of alternate secondary structures. We introduce the formal concept of *RNA multi-structures*, that represent a set of alternate RNA secondary structures in a compact and non redundant way. The definition uses formal grammars. Formal grammars have been applied extensively to the problem of RNA folding, aligning and homology searching [6, 8, 10]. Compared to other tools from language theory, such as deterministic finite automata and HMMs, they are the method of choice because they are able to model long-range interactions. Here we exploit the power of this formalism to encode alternative foldings. The paper is organized as follows. In Section 2, we briefly recall some basic definitions. Section 3 introduces RNA multi-structures. Section 4 presents some examples to illustrate the utility of the concept. Lastly, in Section 5, we describe an algorithm to search for a RNA multi-structure in a genomic sequence.

DOI: 10.15455/CMSR.2014.0002

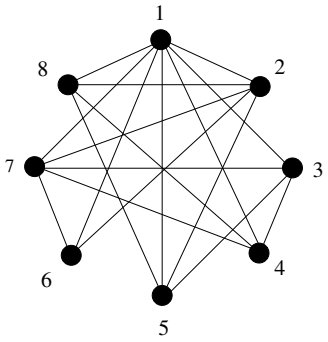
2 RNA secondary structures

Throughout this paper, we see RNA secondary structures as pure combinatorial objects, composed of *helices*. Assume we have a linear sequence S provided with totally ordered positions. An *interval* is a pair of positions $\langle x,y\rangle$ of S such that $x \leq y$. The interval $\langle x,y\rangle$ *precedes* the interval $\langle x',y'\rangle$ if $y < x'$. A *helix* h is a pair of intervals (h_1,h_2) such that $h_1 = \langle x,y\rangle$ precedes $h_2 = \langle x',y'\rangle$. So a helix is characterized by four positions on S, denoted $h.5start$, $h.5end$, $h.3start$, $h.3end$ for x,y,x' and y' respectively. We say that the helix g is *nested* in the helix h if $h.5end < g.5start$ and $g.3end < h.3start$. We say g is *juxtaposed* with h if $g.3end < h.5start$. A *secondary structure* is a set of helices that are all pairwise nested or juxtaposed. Note that in general any relation between any two helices is allowed: They can overlap, form a pseudoknot, or be included in each other.

(a) Set of helices

```
  GUAAAUAUAGUUUAACCAAAACAUCAGAUUGUGAAUCUGACAACAGAGGCUUACGACCCCUUAUUUACC
1 (((((((.............................................................)))))))
2 .........((((.....))))...............................................
3 .........(((((((((.............................))).))))..............
4 ....................(((......)))...................................
5 ......................(((((((...)))))))...........................
6 ............................((((((.............))))))..............
7 ....................................((((....))))....................
8 ........................................((.((.......)).))..........
```

(b) Helix graph



(c) Directed helix graph

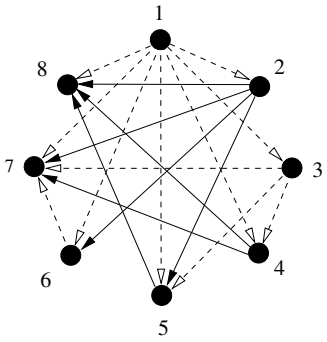


Fig. 1. Human mitochondrial tRNA sequence (source FJ004809.1/12137-12205 – RFAM, RF0005 [3]). (a) Helices are numbered from 1 to 8 from 5' to 3', and are written in Vienna bracket-dot format. (b) Helix graph. (c) Directed helix graph: $h \longrightarrow g$ means that g is juxtaposed with h, and $h - \triangleright g$ means that g is nested in h.

3 RNA multi-structures

Before proceeding to the formal definition, we illustrate our motivation with a simple example taken from the Human mitochondrial tRNA sequence. This sequence is 69nt long and has eight thermodynamically stable helices, which are represented in Figure 1-(a). We denote this set of helices $\mathscr{H}_0$. The elements of $\mathscr{H}_0$ can combine to form a variety of secondary structures. The question that we want to address is: How can we encode this set of all secondary structures, or a given subset of these secondary structures, into a single compact data structure? Furthermore we want to take advantage of the redundancy between the structures, since the structures may share some structural modules, and we want that the data structure can be efficiently used for further queries.

In first approach, the set of helices can be represented by a graph, called the *helix graph*, whose vertices are helices, and there is an edge between every two vertices if helices are either juxtaposed, or nested. This graph is represented in Figure 1-(b). In this graph, the set of all secondary structures is exactly the set of cliques. However, this representation does not convey the necessary information to understand the topology of the set of helices. A better approach is to classify edges in two categories. This is what is done in the directed graph of Figure 1-(c). Plain arcs are for juxtaposed helices, and dotted arcs are for nested helices. Arcs are oriented in the same direction as the *3start* positions on the underlying sequence S. In this graph, any secondary structure can be seen as a selection of plain paths, such that any two plain paths are linked by dotted paths. Each plain path characterizes a *flat structure*.

Definition 1. *A set of helices w is a* flat structure *if, for any two distinct helices h and g in w, h and g are juxtaposed.*

It is clear that any secondary structure can be partitioned into flat structures. We say that a flat structure w is *nested* into a helix h if all helices of w are nested into h. We define a *multi-structure* as a combination of flat structures, generated by a tree grammar.

Definition 2. *A* multi-structure *is a pair $\mathscr{M} = (\mathscr{H}, \mathscr{G})$, where $\mathscr{H}$ is a set of helices and $\mathscr{G}$ is a tree grammar. The grammar alphabet contains a start symbol S, a binary terminal symbol $\circ$, and for each helix h of $\mathscr{H}$, a non-terminal symbol H and a terminal symbol h. All its productions are of the form*

$$\begin{array}{ll} (1) & S \to H_1 \circ \ldots \circ H_q \\ (2) & H \to h(H_1 \circ \ldots \circ H_q) \\ (3) & H \to h \end{array}$$

where $h_1 \circ \ldots \circ h_q$ is any flat structure on $\mathscr{H}$ in (1), and a flat structure that is nested in h in (2).

In the definition of $\mathscr{G}$, $H_1 \circ \ldots \circ H_q$ indicates which plain paths from the helix graph are authorized in the multi-structure, and $h(\ldots)$ which dotted paths are authorized in the multi-structure. The $\circ$ symbol allows to generate trees of arbitrary arity. An *instance* of the multi-structure $\mathscr{M}$ is any structure recognized by its grammar $\mathscr{G}$. It is straightforward to verify that each instance of a multi-structure is a secondary structure.

DOI: 10.15455/CMSR.2014.0002

4 Examples of RNA multi-structures

4.1 Example on mitochondrial tRNA (cont'd)

We continue with our motivating example with Human mitochondrial tRNA sequence introduced in Figure 1. If we want to have *all possible secondary structures* for the helix set $\mathscr{H}_0$, productions for $\mathscr{G}_0$ are all possible rules that respect conditions (1), (2) or (3) in Definition 2.

$$
\begin{aligned}
S \rightarrow\ & H_1 \mid H_2 \mid H_2 \circ H_5 \mid H_2 \circ H_5 \circ H_8 \mid H_2 \circ H_6 \mid H_2 \circ H_7 \mid H_2 \circ H_8 \mid H_3 \mid H_4 \\
& \mid H_4 \circ H_7 \mid H_4 \circ H_8 \mid H_5 \mid H_5 \circ H_8 \mid H_6 \mid H_7 \mid H_8 \\
H_1 \rightarrow\ & h_1(H_2) \mid h_1(H_2 \circ H_5) \mid h_1(H_2 \circ H_5 \circ H_8) \mid h_1(H_2 \circ H_6) \mid h_1(H_2 \circ H_7) \\
& \mid h_1(H_2 \circ H_8) \mid h_1(H_3) \mid h_1(H_4) \mid h_1(H_4 \circ H_7) \mid h_1(H_4 \circ H_8) \\
& \mid h_1(H_5) \mid h_1(H_5 \circ H_8 \circ H_6) \mid h_1(H_7) \mid h_1(H_8) \mid h_1 \\
H_2 \rightarrow\ & h_2 \\
H_3 \rightarrow\ & h_3(H_4) \mid h_3(H_4 \circ H_7) \mid h_3(H_5) \mid h_3(H_7) \mid h_3 \\
H_4 \rightarrow\ & h_4 \\
H_5 \rightarrow\ & h_5 \\
H_6 \rightarrow\ & h_6(H_7) \mid h_6 \\
H_7 \rightarrow\ & h_7 \\
H_8 \rightarrow\ & h_8
\end{aligned}
$$

By construction, the language of this grammar is exactly the set of all secondary structures on $\mathscr{H}_0$. The reader is invited to check that it contains 43 structures.

We take this example a step further and address the problem of building a multi-structure for all *locally optimal secondary structures* on $\mathscr{H}_0$. A locally optimal secondary structures is a secondary structure to which it is not possible to add a supplementary helix [4, 12]. In other words, it is a maximal clique in our helix graph of Figure 1. In [12], we have proved that locally optimal secondary structures are exactly the secondary structures that are partitioned into some *maximal* flat structures (called maximal for juxtaposition structures), corresponding to some maximal plain paths in the helix graph. Here this result allows to define $\mathscr{G}_1$.

$$
\begin{aligned}
S &\rightarrow H_1 \\
H_1 &\rightarrow h_1(H_2 \circ H_5 \circ H_8) \mid h_1(H_2 \circ H_6) \mid h_1(H_3) \mid h_1(H_4 \circ H_8) \\
H_2 &\rightarrow h_2 \\
H_3 &\rightarrow h_3(H_4 \circ H_7) \mid h_3(H_5)
\end{aligned}
\qquad
\begin{aligned}
H_4 &\rightarrow h_4 \\
H_5 &\rightarrow h_5 \\
H_6 &\rightarrow h_6(H_7) \\
H_7 &\rightarrow h_7 \\
H_8 &\rightarrow h_8
\end{aligned}
$$

The number of instances is reduced to 5: $h_1(h_2 \circ h_5 \circ h_8)$, $h_1(h_2 \circ h_6(h_7))$, $h_1(h_3(h_4 \circ h_7))$, $h_1(h_3(h_5))$, $h_1(h_4 \circ h_8)$. These instances correspond to all maximal cliques in the helix graph: $\{1,2,5,8\}, \{1,2,6,7\}, \{1,3,4,7\}, \{1,3,5\}, \{1,4,8\}$. As expected, none are included in each other.

Lastly, multi-structures can be used to encode a *pre-defined set of secondary structures*. This time, we start from a set of candidate secondary structures with low free energy level. We ran the mfold software with 20% suboptimality [13] on our tRNA sequence, and get four suboptimal structures, shown in Figure 2. From this set of structures, we extracted the set of helices, that appears to be the same helix set as $\mathscr{H}_0$. To

Searching for alternate RNA structures in genomic sequences

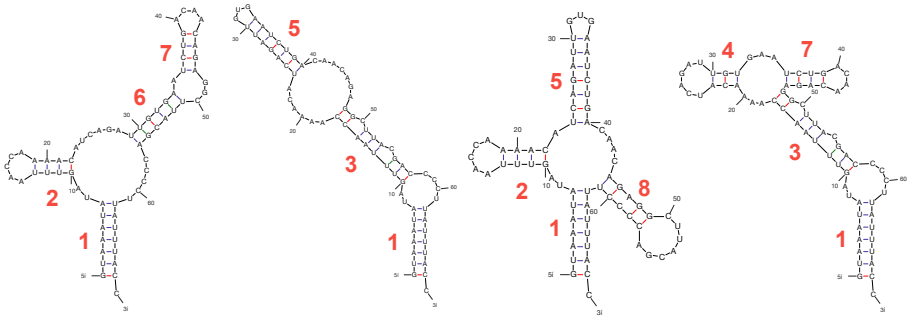


Fig. 2. Mfold output for the Human mitochondrial tRNA sequence

build the productions of the grammar $\mathscr{G}_2$, we consider for each helix h the flat structure composed of all helices connected to the multibranch loop closed by h.

$$
\begin{array}{l}
S \to H_1 \\
H_1 \to h_1(H_2 \circ H_5 \circ H_8) \mid h_1(H_2 \circ H_6) \mid h_1(H_3) \\
H_2 \to h_2 \\
H_3 \to h_3(H_4 \circ H_7) \mid h_3(H_5)
\end{array}
\qquad
\begin{array}{l}
H_4 \to h_4 \\
H_5 \to h_5 \\
H_6 \to h_6(H_7) \\
H_7 \to h_7 \\
H_8 \to h_8
\end{array}
$$

Compared to $\mathscr{G}_1$, the production $H_1 \to h_1(H_4 \circ H_8)$ is missing since none of the four suboptimal structures contains this combination of helices. This new multi-structure has four distinct instances, that correspond exactly to the four suboptimal secondary structures of Figure 2: $\{1,2,6,7\}$, $\{1,3,5\}$, $\{1,2,5,8\}$ and $\{1,3,4,7\}$. This construction is guaranteed to give all input structures. We will see in the following example that it can happen that some new instances are created by the grammar.

4.2 Multi-structures for bacterial RNAse P RNAs

In this second example, we explain how to model an RNA family with a multi-structure. Bacterial RNase P RNAs fall into two major classes that share a common catalytic core, but also show some distinct structural modules: type A (represented by *Escherichia coli* in Figure 3) is the common and ancestral form found in most bacteria, and type B (represented by *Bacillus subtilis* in Figure 3) is found in the low-GC content gram-positive bacteria [7]. We explain how to gather the two structures in a same multi-structure. Following the nomenclature of the RNase P database, helices are labeled $P_1 - P_{18}$ from 5' to 3' in the *E. coli* structure. Helices P_4 and P_6 form pseudoknots and do not belong to the secondary structure. *B. subtilis* features some additional helices, denoted $P_{5.1}$, $P_{10.1}$, $P_{15.1}$ and P_{19}, and lacks helices P_{13}, P_{14}, P_{16} and P_{17}. The corresponding grammar is obtained by taking all flat structures of each of the two consensus secondary structures for type A and type B.

$$
\begin{aligned}
S &\to P_1 \\
P_1 &\to p_1(P_2) \mid p_1(P_2 \circ P_{19}) \\
P_2 &\to p_2(P_3) \mid p_2(P_3 \circ P_5 \circ P_{15} \circ P_{18}) \mid p_2(P_3 \circ P_5 \circ P_{15} \circ P_{15.1} \circ P_{18}) \\
P_3 &\to p_3 \\
P_5 &\to p_5(P_{5.1} \circ P_7) \mid p_5(P_7) \\
P_{5.1} &\to p_{5.1} \\
P_7 &\to p_7(P_8 \circ P_9 \circ P_{10}) \\
P_8 &\to p_8 \\
P_9 &\to p_9 \\
P_{10} &\to p_{10}(P_{10.1} \circ P_{11}) \mid p_{11} \\
P_{10.1} &\to p_{10.1} \\
P_{11} &\to p_{11}(P_{12}) \mid p_{11}(P_{12} \circ P_{13} \circ P_{14}) \\
P_{12} &\to p_{12} \\
P_{13} &\to p_{13} \\
P_{14} &\to p_{14} \\
P_{15} &\to p_{15}(P_{16}) \mid p_{15} \\
P_{15.1} &\to p_{15.1} \\
P_{16} &\to p_{16}(P_{17}) \\
P_{17} &\to p_{17} \\
P_{18} &\to p_{18} \\
P_{19} &\to p_{19}
\end{aligned}
$$

Green non-sulfur Bacteria RNase P RNAs are known to show some notable variation against the forms A and B. The majority of them (represented here by *H. auriantacus* in Figure 3) are of the type A class, except for the structural module $P_{18}/P_{15.1}$ that is instead quite similar to that of the type B and for the helix P_{19}. One exception is represented by the sequence of *T. roseum*, which has independently converged with the class B RNAs (presence of helices $P_{10.1}$ and $P_{15.1}$, and absence of helices P_{13}, P_{14} and P_{19}). However, *T. roseum* RNA retains P_{16}/P_{17} and does not contain $P_{5.1}$ [7] (see Figure 3). Interestingly, these two variants appear as instances of the multi-structure designed to encode solely types A and B. It means that in this example, these two new structures created by the multi-structure are functional and have been selected by the evolution.

5 Searching multi-structures in genomic sequences

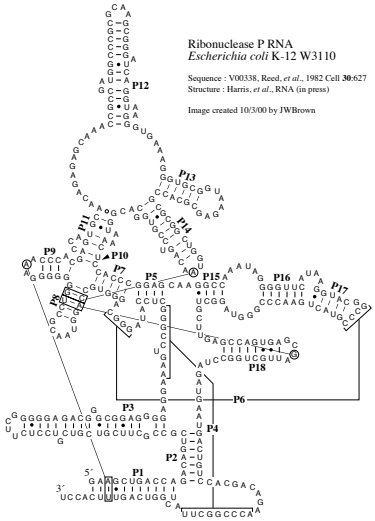
We now turn to the following problem: Given a multi-structure $\mathscr{M} = (\mathscr{H}, \mathscr{G})$ and a text such as a genomic sequence, how to identify occurrences of $\mathscr{M}$ in a text. For that, we assume that the text is given by a list of putative helices K, that have been obtained by preprocessing the genomic sequence. We define formally what is an *occurrence* of a multi-structure in such a text.

Definition 3. *Let $\mathscr{H}$ and K be two sets of helices. A* helix mapping *from $\mathscr{H}$ to K is an injective function $\phi : \mathscr{H} \mapsto K$ such that, for any two helices h and h' in $\mathscr{H}$, if h is nested in h', then $\phi(h)$ is nested in $\phi(h')$, and if h is juxtaposed with h', then $\phi(h)$ is juxtaposed with $\phi(h')$.*

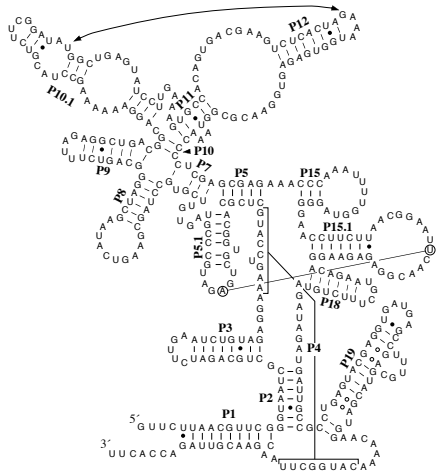
Definition 4. *Let $\mathscr{M} = (\mathscr{H}, \mathscr{G})$ be a multi-structure and K be a text defined by a set of helices. An* occurrence *of $\mathscr{M}$ in K is a pair $(\mathscr{H}', \phi)$, where $\mathscr{H}'$ is a subset of $\mathscr{H}$ and ϕ is a helix mapping from $\mathscr{H}'$ to K.*

DOI: 10.15455/CMSR.2014.0002

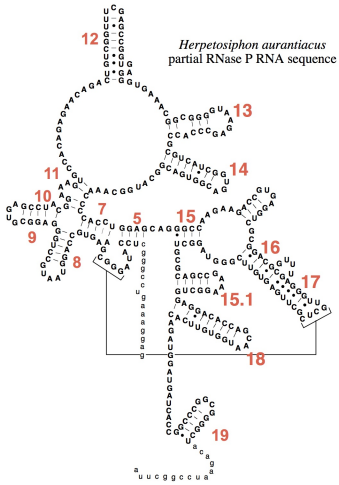
Searching for alternate RNA structures in genomic sequences



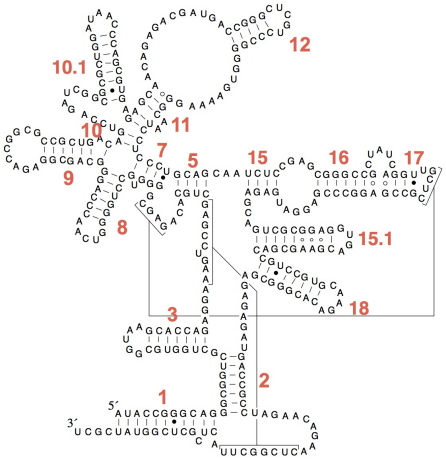
Type A: *E. coli*



Type B: *B. subtilis*



H. aurantiacus, partial sequence



T. roseum

Fig. 3. Secondary structures for bacterial RNase P RNA (source [2]).

DOI: 10.15455/CMSR.2014.0002

This definition allows for approximate search with errors, since $\mathscr{H}'$ is not required to contain all helices of the instances of the multi-structure. Note also that in this general definition, we do not specify whether one instance or all instances of the multi-structure should appear in the text. This gives rise to two distinct searching problems, that are formalized as follows.

Simple occurrence problem: Let e be a natural number. Find in the text K all occurrences of $\mathscr{M}$ with at most e errors, where the number of errors of an occurrence $(\mathscr{H}', \phi)$ is defined as the *minimal* number of helices appearing in some instance of $\mathscr{M}$ and that are not in $\mathscr{H}'$.

Universal occurrence problem: Let e be a natural number. Find in the text K all occurrences of $\mathscr{M}$ with at most e errors, where the number of errors of an occurrence $(\mathscr{H}', \phi)$ is defined as the *maximal* number of helices appearing in some instance of $\mathscr{M}$ and that are not in $\mathscr{H}'$.

The simple occurrence problem consists in finding in the text K all positions where at least *one* possible instance of the multi-structure matches the putative helices K. For example, considering the multi-structure $\mathscr{M} = (\mathscr{H}_0, \mathscr{G}_2)$, the occurrence $(\{1,3,4,5\}, \phi)$, with $\phi(1) = 1$, $\phi(3) = 3$, $\phi(4) = 5$ and $\phi(5) = 7$, has no errors because it matches the instance $\{1,3,5\}$. This applies typically to RNA sequences that are provided with a set of potential secondary structures, such as suboptimal secondary structures of Figure 2, or variants for bacterial RNase P RNA of Figure 3. When the multi-structure encodes a set of suboptimal predicted structures, this can serve for example to improve homology searching, even if the real structure is not know: Build the associated multi-structure, then use this multi-structure to scan a genomic sequence in conjunction with sequence similarity.

The universal occurrence problem asks that *all* instances of the multi-structure match the same location in the text. This applies for example to RNA sequences that support several stable folding states, such as riboswitches.

We need some preliminary definitions to describe the algorithms for the simple occurrence and universal occurrence problems. We first define two total orders between helices, that apply both on helices of the multi-structure and on helices of the text. $\preccurlyeq$ is such that the start positions of the helices are sorted from 5' to 3': $f \preccurlyeq g \iff f.5start \leq g.5start$, and orders arbitrarily helices with the same *5start* position. $\sqsubseteq$ is such that the end positions of the helices are sorted from 5' to 3': $f \sqsubseteq g \iff f.3end \leq g.3end$, and orders arbitrarily helices with the same *3end* position. In the text K, we denote $K[k..\ell]$ the set of helices that are greater than k for the $\preccurlyeq$ ordering and smaller than ℓ for the $\sqsubseteq$ ordering: $K[k..\ell] = \{f \in K; k \preccurlyeq f \sqsubseteq \ell\}$.

The algorithm works by decomposing the multi-structure into smaller structures. Given a flat structure $w = h_1 \circ \ldots \circ h_q$, $\mathscr{M}[h_1 \circ \ldots \circ h_q]$ denotes the structures that originate from $h_1 \circ \ldots \circ h_q$: All structures that are generated from $H_1 \circ \ldots \circ H_q$ in $\mathscr{G}$.

We define $S(w,k,\ell)$ as the minimal number of errors to match $\mathscr{M}[w]$ with text helices in $K[k..\ell]$ in the *simple occurrence problem*, and $T(h,k,\ell)$ as the minimal number of errors to matches structures nested in h with text helices in $K[k\ldots\ell]$:

$$T(h,k,l) = \boxed{\min}\{S(w,k,\ell), H \to h(w) \in \mathscr{G})\} \qquad (\diamond)$$

Equations for calculating S are given in Figure 4.

DOI: 10.15455/CMSR.2014.0002

Searching for alternate RNA structures in genomic sequences

$$S(\lambda, k, \ell) = 0$$

$$S(h_1 \circ h_2 \circ \ldots \circ h_q, k, \ell) =$$

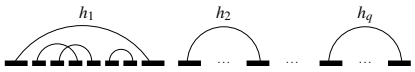
Case 1: if $K[k..\ell]$ is empty, then $||\mathscr{M}[h_1 \circ \ldots \circ h_q]||$

Case 2: otherwise, if $\ell \sqsubset k$, then $S(h_1 \circ \ldots \circ h_q, k+1, \ell)$

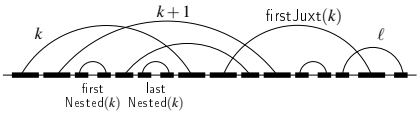
Case 3: otherwise

$$\min \begin{cases} \textit{3-a: Helix k of the text is not in the multi-structure} \\ S(h_1 \circ \ldots \circ h_q, k+1, \ell) \\ \textit{3-b: Helix } h_1 \textit{ is not present in the text, but some helix of } \mathscr{M}[h_1] \\ \textit{is found in the text} \\ \min\limits_{k \preccurlyeq p \sqsubseteq \ell} 1 + T(h_1, k, p) + S(h_2 \circ \ldots \circ h_q, \mathsf{firstJuxt}(p), \ell) \\ \textit{3-c: No helix of } \mathscr{M}[h_1] \textit{ is found in the text} \\ ||\mathscr{M}[h_1]|| + S(h_2 \circ \ldots \circ h_q, k, \ell) \\ \textit{3-d: Helix } h_1 \textit{ is matched with helix k of the text} \\ T(h_1, \mathsf{firstNested}(k), \mathsf{lastNested}(k)) + S(h_2 \circ \ldots \circ h_q, \mathsf{firstJuxt}(k), \ell) \end{cases}$$

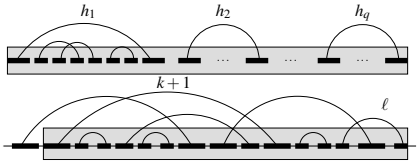
Multi-structure



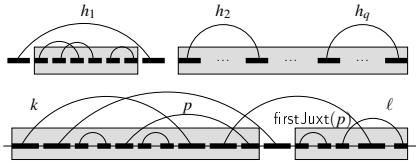
Text



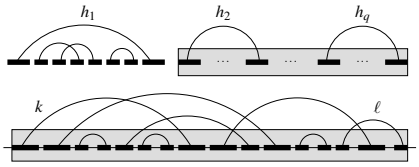
Case 3-a: Helix k is not in the multi-structure



Case 3-b: Helix h_1 is not in the text but some helix of $\mathscr{M}[h_1]$ is



Case 3-c: No helix of $\mathscr{M}[h_1]$ is found in the text



Case 3-d: Helix h_1 is matched with k

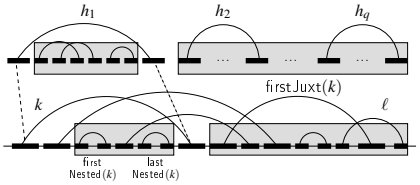


Fig. 4. Recurrence equations to compute S values for the *simple occurrence problem*. λ denotes the empty flat structure, $k+1$ the helix succeeding k for $\preccurlyeq$, $\mathsf{firstJuxt}(k)$ the first helix according to $\preccurlyeq$ that is juxtaposed with k, $\mathsf{firstNested}(k)$ the first helix according to $\preccurlyeq$ that is nested in k, and $\mathsf{lastNested}(k)$ the last helix according to $\sqsubseteq$ that is nested in k. $||\mathscr{M}[w]|| 1 + \boxed{\min}\{||\mathscr{M}[w]||; H \rightarrow h(w) \in \mathscr{G}\}$ and $||\mathscr{M}[h_1 \circ \ldots \circ h_q]|| = ||\mathscr{M}[h_1]|| + ||\mathscr{M}[h_2 \circ \ldots \circ h_q]||$. Cases 1 and 2 are initial cases. We give below an example for case 3.

DOI: 10.15455/CMSR.2014.0002

Property 1: The value $\min\{S(w,k_0,\ell_0), S \to w \in \mathscr{G}\}$, where k_0 is the smallest helix of K for the $\preccurlyeq$ ordering and ℓ_0 the largest helix of K for the $\sqsubseteq$ ordering, gives the number of errors to match $\mathscr{M}$ against K, such as defined in the *simple occurrence problem*.

For the *universal occurrence problem*, analogous equations hold, where the two $\boxed{\min}$ operators in Equation $(\diamond)$ and in the caption of Figure 4 have to be replaced by max operators when computing T and $||\mathscr{M}[w]||$. The number of errors to match $\mathscr{M}$ against K is then given by $\max\{S(w,k_0,\ell_0), S \to w \in \mathscr{G}\}$.

```
Initialization of all values in T to +∞
For each helix ℓ in the text K in increasing order for ⊑
  w_old := λ
  For each flat structure w = h_1 ∘ ... ∘ h_q in W in increasing order for ⊑_lex
    s := length of the longest common suffix between w and w_old
    For each helix k in K[1..ℓ] in decreasing order for ≼
      For i := s + 1 to q
        compute S'[i,k] := S(h_{q-(i-1)} ∘ ... ∘ h_q, k, ℓ) with Equations of Fig. 4   (⋆)
      End for i
      For each helix h in H such that H → h(w) ∈ G
        T[h,k,ℓ] := min(T[h,k,ℓ], S'[q-1,k])
      End for h
    End for k
    w_old := w
  End for w
End for ℓ
```

Fig. 5. Implementation of equations of Figure 4

It is possible to implement equations of Figure 4 using dynamic programming for T and S. There are mn^2 values to compute for function T, where m and n are the numbers of helices in $\mathscr{H}$ and K respectively. Considering S, one can note that the first parameter w is always a *suffix* of some flat structure. A *suffix* of $w = h_1 \circ \ldots \circ h_q$ is any flat structure of the form $h_i \circ \cdots \circ h_q$ for any $1 \leq i \leq q$. If we denote by W the set of flat structures on $\mathscr{H}$ that appears in a right-hand-side of a production of $\mathscr{G}$, the total number of all different suffixes of W is bounded by $\sigma = \sum_{w \in W} |w|$. So the total number of different values to compute for S and T is in $O(mn^2 + \sigma n^2)$. We still have to discuss the order of execution to compute T and S. When computing $T(h,k,\ell)$, we need to access values of S of the form $S(w,k,\ell)$, where w is a flat structure nested in h. When computing $S(w,k,\ell)$, and supposing that all required values for T are known, we only need to access values of S that are of the form $S(w',k',\ell)$, where w' is always a suffix of w. More precisely, either $w' = w$ and $k \prec k'$, or w' is a proper suffix of w and $k \preccurlyeq k'$. Note also that the last parameter ℓ is unchanged. This means that to compute $S(w,k,\ell)$, we can forget all previously computed values for S whose first parameter is not a suffix of w and whose last parameter is not ℓ. However, as some flat structures of W may share common suffixes, some $S(w',k',\ell)$ values are used several times. To prevent redundant useless computations, it is thus necessary to order flat structures according to their suffixes. We

DOI: 10.15455/CMSR.2014.0002

define the $\sqsubseteq_{\mathsf{lex}}$ order on W as the lexicographic ordering built on $\sqsubseteq$, starting from the rightmost helices: $h_1 \circ \cdots \circ h_{q-1} \circ h_q \sqsubseteq_{\mathsf{lex}} h'_1 \circ \cdots \circ h'_{q'-1} \circ h'_{q'}$ if, and only if, $h_q \sqsubset h'_{q'}$, or $h_q = h'_{q'}$ and $h_1 \circ \cdots \circ h_{q-1} \sqsubseteq_{\mathsf{lex}} h'_1 \circ \cdots \circ h'_{q'-1}$.

It follows from these remarks that helices ℓ should be enumerated in increasing order for $\sqsubseteq$, flat structures w of W in increasing order for the lexicographic ordering $\sqsubseteq_{\mathsf{lex}}$, helices k in decreasing order for $\preccurlyeq$, and suffixes of a given flat structure in decreasing order for $\preccurlyeq$. Figure 5 shows the resulting algorithm, using a permanent two-dimensional table for T of size mn^2 and a temporary two-dimensional table S' for S. For a given helix ℓ and a flat structure w, we eventually store in $S'[j,k]$ the value of S for the suffix of w of size j, compared to the text $K[k..\ell]$, that is $S'[j,k] = S(h_{q-(j-1)} \circ \ldots \circ h_q, k, \ell)$. The table S' is of size $y \times n$, where $y \leq m$ is the maximal length of a flat structure in W, and n is the number of helices in K. This algorithm thus requires space $O(mn^2)$. As for the time complexity, the loop on ℓ has $O(n)$ iterations, and, for each ℓ and w, the loop on k has also $O(n)$ iterations. In the worst case, there is no common suffixes between flat structures of W: for each flat structure w, the loop on i covers all helices of w, in $O(y)$ executions of the $(\star)$ line. For any fixed ℓ and k, the total number of executions of the $(\star)$ line is thus bounded by σ. Finally, each execution of the $(\star)$ line takes at worst $O(n)$ time (loop on p with $k \preccurlyeq p \sqsubseteq \ell$, see case 3b of Figure 4). Taking all together, the algorithm runs is in $O(\sigma n^3)$ worst-case time. Note that the algorithm of Figure 5 computes the minimal number of errors. Retrieving the actual best alignments of the pattern with the text is done through backtracking in the T table, recomputing only the relevant S tables.

6 Conclusion

We have shown that our definition of RNA multi-structure can faithfully model several situations where alternate structures naturally arise. This result suggests many questions for future research: How to build automatically multi-structures? How to sample or enumerate them? How to compare them? How to search for structural elements in a multi-structure? In this paper, we have investigated the pattern matching problem, and have provided an algorithm to search for RNA multi-structures in a text. We have made the decision to stay at a abstract level. The RNA sequence and the text are both defined as sets of helices, that can be either nested or juxtaposed. However, there are several directions to make the model more realistic, such as adding compatibility relations or distance contraints between helices. This additional information also makes the algorithm significantly faster in practice. A prototype has been implemented and is available upon request.

Acknowledgments: The authors would like to thank Robert Giegerich (Bielefeld Universität) for fruitful discussions and valuable suggestions, which helped to improve significantly this work.

References

1. C.A. Brosnan and O. Voinnet. The long and the short of noncoding RNAs. *Current Opinion in Cell Biology*, 21, 2009.

2. J.W. Brown. The Ribonuclease P Database. *Nucleic Acids Research*, 27(1):314, 1999.
3. S.W. Burge, J. Daub, R. Eberhardt, J. Tate, L. Barquist, E.P. Nawrocki, S.R. Eddy, P.P. Gardner, and A. Bateman. Rfam 11.0: 10 years of RNA families. *Nucleic Acids Research*, 2012.
4. P. Clote. An efficient algorithm to compute the landscape of locally optimal RNA secondary structures with respect to the Nussinov-Jacobson energy model. *J. Computational Biology*, 1:83–101, 2005.
5. A. L. Edwards and R. T Batey. Riboswitches: A common RNA regulatory element. *Nature Education*, 3(9):9, 2010.
6. R. Giegerich and H. Touzet. Modeling dynamic programming problems over sequences and trees with inverse coupled rewrite systems. *Algorithms*, 7(1):62–144, 2014.
7. E.S. Haas and Brown JW. Evolutionary variation in bacterial RNase P RNAs. *Nucleic Acids Res.*, 26(18):4093–9, 1998.
8. I. Hofacker and P. Stadler. Product grammars for alignment and folding. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 99, 2014.
9. J.S. Mattick and I.V. Makunin. Non-coding RNA. *Hum Mol Genet*, 15 Spec No 1:R17–29, 2006.
10. E.P. Nawrocki, D.L. Kolbe, and S.R. Eddy. Infernal 1.0: Inference of RNA alignments. *Bioinformatics*, 25:1335–1337, 2009.
11. J. Ritz, J. S. Martin, and A. Laederach. Evolutionary evidence for alternative structure in RNA sequence co-variation. *PLoS Comput Biol*, 9(7):e1003152, 2013.
12. A. Saffarian, M. Giraud, A. de Monte, and H. Touzet. RNA Locally Optimal Secondary Structures. *Journal of Computational Biology*, pages 1120–1133, 2012.
13. M. Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*, 31(13):3406–3415, 2003.

DOI: 10.15455/CMSR.2014.0002