Journal of Bioinformatics and Computational Biology

LOCAL SEQUENCE-STRUCTURE MOTIFS IN RNA

ROLF BACKOFEN and SEBASTIAN WILL

Chair for Bioinformatics at the Institute of Computer Science,
Friedrich-Schiller-Universitaet Jena, Ernst-Abbe-Platz 2,
D-07743 Jena, Germany,
{*backofen,will*}*@inf.uni-jena.de*

RNA enjoys increasing interest in molecular biology; despite this interest fundamental algorithms are lacking, e.g. for identifying local motifs. As proteins, RNA molecules have a distinctive structure. Therefore, in addition to sequence information, structure plays an important part in assessing the similarity of RNAs. Furthermore, common sequence-structure features in two or several RNA molecules are often only spatially local, where possibly large parts of the molecules are dissimilar. Consequently, we address the problem of comparing RNA molecules by computing an optimal local alignment with respect to sequence *and* structure information. While local alignment is superior to global alignment for identifying local similarities, no general local sequence-structure alignment algorithms are currently known. We suggest a new general definition of locality for sequence-structure alignments that is biologically motivated *and* efficiently tractable. To show the former, we discuss locality of RNA and prove that the defined locality means connectivity by atomic and non-atomic bonds. To show the latter, we present an efficient algorithm for the newly defined pairwise local sequence-structure alignment (lssa) problem for RNA. For molecules of lengthes n and m, the algorithm has worst-case time complexity of $O(n^2 \cdot m^2 \cdot \max(n,m))$ and a space complexity of only $O(n \cdot m)$. An implementation of our algorithm is available at *http://www.bio.inf.uni-jena.de*. Its runtime is competitive with global sequence-structure alignment.

Keywords: RNA; local alignment; local sequence-structure alignment; lssa.

1. Introduction

The role of ribonucleic acid (RNA) in biological systems was largely underestimated for a long time. Today, RNA enjoys increasing attention due to recent discoveries such as the existence of small RNAs, which are strongly involved in cell control (see Cousin[1]). Despite increasing interest in RNA, the important problem of comparing RNA molecules in order to identify local motifs is still unsolved; even its formal understanding is unsatisfying.

We illustrate this problem by means of a typical scenario, where one has to compare RNA molecules. Suppose we are interested in the RNA motif that binds to a certain protein. For example, we want to identify the mRNA element SECIS (short for **SE**leno**C**ysteine **I**nsertion **S**equence) as investigated e.g. by Wilting *et al.*[2], which binds to the protein SelB. By certain means, e.g. by SELEX (see Klug and Famulok[3]) we get a pool of RNAs that contain the motif of interest.

However, for our approach neither the origin of the set of RNAs nor the kind of the binding motif is limited. Anyway, given a set of RNA molecules, we are left with de-

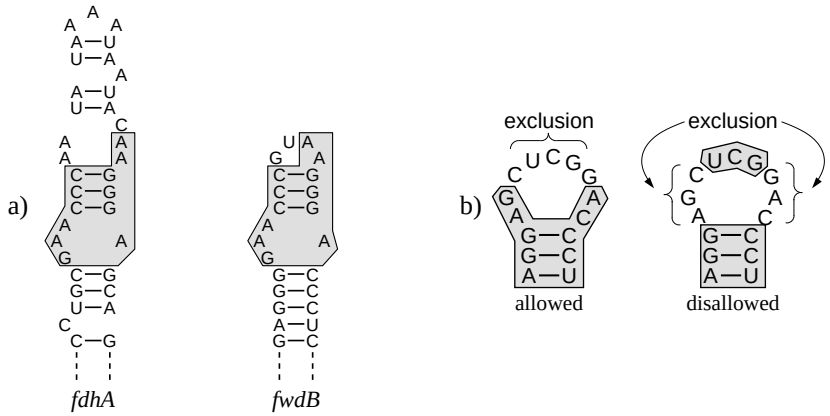

Fig. 1. a) Putative SECIS-motif (see Wilting *et al.*[2]). The identical bases, which form the minimal local motif, are highlighted. b) Allowed vs. disallowed exclusions.

termining the similar regions in them. Whereas in such cases today's biologists compare the RNA molecules manually, we identify the similarities automatically. To this end, we introduce the local sequence-structure alignment (lssa); the local comparison of RNAs is complicated by two particularities.

First, in our example the protein binds to the RNA due to its sequence and structure. Consequently, the similarity of the bound RNAs is based on sequence and structure. In general, RNA molecules have a distinct and complex three-dimensional structure, due to the single strand occurrence of RNA. Sequence *and* structure of RNA molecules are biologically meaningful and thus conserved in evolution. It is therefore essential to compare RNA considering both features. Second, as it is the case for the SECIS motif, often similarities between two or several molecules are only local, i.e. some parts of the molecules share great similarity, whereas other parts are unrelated.

In contrast to the first aspect, which is also subject of the recently developed global sequence-structure alignment algorithms, the second aspect even lacks proper understanding up to now, which we will elucidate in the following subsection.

1.1. *A Note on Locality*

Since the meaning of locality is more intricate in the context of sequence-structure alignment than of pure sequence alignment, we explain our notion in analogy to sequence alignment. A local alignment of sequences is commonly defined as a (global) alignment of one pair of subsequences of the input sequences. Note that the bases in a subsequence are connected via the backbone, which constitutes a dependency. For RNA, several definitions of local alignment are possible. If we define the local alignment again as the best alignment of subsequences, we ignore the RNA structure completely. Hence, in a next step we require that the subsequences represent complete substructures. (Later we are going to introduce the term arc-complete for this property.) This kind of locality is required for an appropri-

ate definition of local sequence-structure alignment. Additionally, one can exclude certain substructures from a substructure, while the spatial locality is preserved due to connection of bases by non-atomic H-bonds, in the following called *arcs*. Our algorithm handles this form of locality, i.e. connected substructures with excluded substructures. The small example in Fig. 1a shows that this indeed is the preferable notion of locality. The figure shows the putative SECIS-elements in the archaea *Methanococcus jannaschii* proposed by Wilting *et al*.[2] (see Fig. 1a, where the putative motif is boxed). Since the apical subsequence AAUAUAAAAUAAUAC in the left molecule *fdhA* has no correspondence in *fwdB*, a correct local alignment of the two RNAs aligns two pairs of subsequences, which are isolated on the sequence level but connected by structure.

This should not be confused with the output of local sequence alignment programs such as BLAST[4], which typically yield several isolated pairs of aligned subsequences. In sequence alignment, these subsequence pairs can be aligned and scored independently and are just the k best non-overlapping local alignments. However, for sequence-structure alignment, the dependency created by the arcs forbids this independent treatment. We will not discuss the analogous extension of sequence-structure alignment for yielding a number of best non-overlapping alignments here.

Furthermore, it is reasonable to consider only conserved arcs for forming a connection of two otherwise isolated subsequences. Conserved arcs are those arcs that are matched by our alignment. In consequence, our locality can be defined only for alignments and not for single RNA sequence-structures. Additionally, we treat an arc as an entity, i.e. we align either both bases of an arc or none.

In order to get exactly those connected subsets of bases as local motifs, we allow at most one exclusion of a substructure in each loop (regardless which kind of loop). Consider Fig. 1b as an example. Whereas the left structure contains one exclusion and is therefore an admissible local motif, the right one contains two exclusions in the same loop, thus producing an unconnected, and hence forbidden motif.

Note that since the exclusion of certain substructures in substructures is algorithmically most challenging, our presentation and the examples given in this paper focus on this aspect of locality.

1.2. *Related Work*

For a proper classification and comparison to related work, we have to discuss RNA structure more in detail. It is common for RNA algorithms to handle only the secondary structure of the molecules, i.e. the set of non-atomic bonds (arcs) between pairs of bases within one molecule. When considering secondary structure, one further differentiates between general structures (aka crossing structures) and the class of nested structures, which forbid pairs of crossing arcs. Often it is only the restriction to nested structures that makes a problem algorithmically tractable. For example, the most prominent RNA structure prediction algorithms compute nested secondary structure (see Zuker[5], Wuchty *et al*.[6]).

Consequently, one distinguishes several RNA alignment problems with different complexity. Namely, these problems are alignment crossing vs. crossing, nested vs. crossing,

nested vs. nested, as well as alignments of structure vs. plain sequence (see Jiang *et al.*[7]). In this paper, we consider local nested vs. nested alignment.

Recently, Jiang *et al.*[7] defined a *global* general edit distance of RNA based on sequence and structure. They give an efficient $O(n^2 \cdot m^2)$ dynamic programming algorithm for *global* alignment of nested against crossing structures for a specialization of their distance score. Furthermore, they show that the global alignment of crossing against plain (and in consequence, nested against crossing) RNA is NP-complete for their general distance score.

Despite NP-completeness, there are approaches to align crossing structures against plain sequence. Lenhof *et al.*[8] give an integer linear programming (ILP) approach to the alignment of RNA structure to plain sequence. Eddy[9] discusses the similar problem of aligning a sequence against a covariance model — a description of an ensemble of RNA sequences and structures.

Furthermore, special cases of *local* alignment of RNAs have been dealt with previously. Gorodkin *et al.*[10] have identified common stem loops. Improving this significantly, Hoechsmann *et al.*[11], who handle RNA alignment by tree alignment based on an earlier work of Jiang *et al.*[12], examine the problem of finding the most similar subtrees. For example, both algorithms cannot identify the motif of the RNAs in Fig. 1a as given in the literature, since in contrast to our approach excluded substructures in a substructure are not considered.[a] The tree alignment approach imposes some restrictions on the sequences of edit operations (and thus the alignments). In order to overcome these restrictions Jiang *et al.*[7] recently developed a general edit distance, which constitutes the basis for our general similarity. For further discussion, see Jiang *et al.*[7] and Hoechsmann *et al.*[11].

Finally, protein structure alignment is closely related to RNA alignment. In principle, if the 3D-structure of the RNAs is known, then methods developed for proteins, as discussed by Gerstein *et al.*[13], are applicable to RNA as well. However, algorithms based on RNA secondary structure are generally more efficient for this purpose. The ILP branch and cut algorithm for protein structure alignment described by Lancia *et al.*[14] uses contact maps of proteins, a representation of structure that is almost equal to crossing secondary structure of RNAs. Therefore, the problem there is closely related to crossing vs. crossing RNA alignment, which is NP-complete. The problem of aligning a plain sequence to a sequence and structure is called protein threading. When using contact maps as done by Xu et al.[15], protein threading is very close to the alignment of plain RNA sequence vs. RNA with crossing structure.

1.3. *Contribution and Plan of the Paper*

We present a sequence-structure alignment method that handles local motifs as shown e.g. in Figs. 1a and 8, where none of the previously mentioned approaches can be applied. Therefore, we define the local sequence-structure alignment (lssa) problem and then formulate an efficient algorithm to solve it.

[a]Hoechsmann *et al.* discuss this issue as well and refer to it as local pattern similarity between trees, which is not handled by their algorithm.

In the subsection on similarity, we develop a general similarity, which is as general as the recently defined edit distance by Jiang *et al.*[7]. A similarity score is a necessary prerequisite for local alignment. A major contribution of this paper is our biologically motivated definition of local alignments. As discussed previously, the notion of locality is by no means trivial in the context of RNA alignment. We translate the biological intuition to a mathematical concept, namely connectivity of a motif graph, and show that our definition of locality is equivalent to this formalization. To our knowledge the definition of locality presented here is the first non-trivial and algorithmically tractable one for RNA.

The discussion of similarity and locality then allows a definition of the lssa problem. Finally, we will show how the lssa problem is efficiently tractable by dynamic programming. We will demonstrate the use of the defined terms and applicability of the algorithm by giving real world examples in the results section.

2. Local Alignment of RNA

2.1. *Preliminaries*

A *sequence* S is a word over the alphabet $\{A,C,G,U\}$, $S[i]$ denotes the ith symbol in S. An *arc* a is a pair $(i,j) \in \mathbb{N} \times \mathbb{N}$, such that $i < j$. i and j are called *ends* of the arc a. We also use the notations a^l and a^r for i and j, respectively.

A *structure* P is a set of arcs, such that no end of an arc appears more than once in P. We call two arcs $(i_1,i'_1),(i_2,i'_2)$ *crossing*, if and only if $i_1 < i_2 < i'_1 < i'_2 \vee i_2 < i_1 < i'_2 < i'_1$. A structure containing at least one pair of crossing arcs is called *crossing*, otherwise it is called *nested*. We call the tuple $\mathcal{S} = (S,P)$ a *sequence-structure*. We impose a partial order $\prec_P$ between arcs by defining $(i_1,i'_1) \prec_P (i_2,i'_2)$ if and only if $i_2 < i_1 < i'_1 < i'_2$.

A *range* $[k..k']$ is the set of positions $\{k,k+1,\ldots,k'\}$. Let $I,J \subset \mathbb{N}$ denote two arbitrary sets of positions. The symbol $-$ denotes a gap. An *alignment A of two sequences S_1 and S_2* is a subset of $[1..|S_1|] \cup \{-\} \times [1..|S_2|] \cup \{-\}$, where for all pairs $(i,j),(i',j') \in A$ holds 1.) $i \leq i' \Rightarrow j \leq j'$ 2.) $i = i' \neq - \Rightarrow j = j'$, and 3.) $j = j' \neq - \Rightarrow i = i'$. Intentionally, unaligned positions in the sequences are allowed, which means that alignments can be local.[b]

In the following, we fix two sequence-structures $\mathcal{S}_1 = (S_1,P_1)$ and $\mathcal{S}_2 = (S_2,P_2)$, where P_1 and P_2 are nested. Given an arc $(i,i') \in P_1$ (resp. $(i,i') \in P_2$), then the arc (i,i') is called *aligned* in the alignment A if and only if there is an arc (j,j') in the other structure P_2 (resp. P_1), such that $(i,j),(i',j') \in A$ (resp. $(j,i),(j',i') \in A$).

I is called *arc-complete for a structure P*, write $\mathrm{ac}_P(I)$, iff for every arc $(i,i') \in P$ holds either $i,i' \in I$ or $i,i' \notin I$. The term arc-complete formalizes our treatment of an arc as an entity. By restricting ourselves to arc-complete sets, we disallow that the two bases of each arc are seperated.

Let $\pi_1(A)$ (resp. $\pi_2(A)$) denote the projection to the aligned positions in the first (resp. second) sequence of A. Then, we call A *arc-complete* for P_1 and P_2, iff $\pi_l(A)$ is arc-complete for P_l for $l \in \{1,2\}$.

[b]In contrast, the alignment A is called *global*, if for every $1 \leq i \leq |S_1|$ there is an edge $(i,j) \in A$ and for every $1 \leq j \leq |S_2|$ there is an edge $(i,j) \in A$.

We define a binary $0/1$-function inc_P, by $\mathrm{inc}_P(i) = 1$ iff there is an arc $a \in P$ ending with i. Since we fixed the structures P_1 and P_2, we will write inc_1 instead of inc_{P_1} in the following. Analogously, we will use the abbreviations inc_2, ac_1, and ac_2.

2.2. *Similarity*

Our similarity function differentiates matches, insertions and deletions of bases without incident arcs as a score for sequence alignment. Furthermore, it scores matches of arcs and breakings of arcs. Here, we have an arc breaking, whenever either at least one end of an arc is aligned to a gap or the two ends are aligned to bases that are not connected by an arc. In contrast, Jiang *et al.*[7] distinguish between arc breaking, arc altering and arc removing, which we handle as sub-cases of arc breaking.

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

Let A be an (arbitrary) alignment of S_1 and S_2. We define the *general similarity score of A given by the functions* s_b, s_{arc}, s_{br1}, *and* s_{br2} *for* $\boldsymbol{\mathcal{S}}_1$ *and* $\boldsymbol{\mathcal{S}}_2$ as

$$\begin{aligned}\textsc{SimScore}(\boldsymbol{\mathcal{S}}_1,\boldsymbol{\mathcal{S}}_2,A) \;=\; & \sum_{\substack{(i,j)\in A\\ \neg\,\mathrm{inc}_1(i)\wedge\neg\,\mathrm{inc}_2(j)}} \mathrm{s_b}(i,j) \;+\; \sum_{\substack{(i,i')\in P_1,(j,j')\in P_2\\ (i,j)\in A,(i',j')\in A}} \mathrm{s_{arc}}((i,j),(i',j')) \\ & +\; \sum_{\substack{(i,i')\in P_1,(j,j')\notin P_2\\ (i,j)\in A,(i',j')\in A}} \mathrm{s_{br1}}((i,i'),j,j') \;+\; \sum_{\substack{(i,i')\notin P_1,(j,j')\in P_2\\ (i,j)\in A,(i',j')\in A}} \mathrm{s_{br2}}(i,i',(j,j')).\end{aligned} \tag{1}$$

According to this definition the similarity score is a sum of base similarities between free bases and other free bases or gaps that are aligned by A, the similarities of arcs matched by A, and the penalties for arc breakings.

Like Jiang *et al.*[7], we use a slightly specialized scoring scheme in the efficient alignment algorithm, for which we will use the term *similarity score*.[c] Therefore, we restrict the functions $\mathrm{s_{br1}}$ and $\mathrm{s_{br2}}$ by

$$\mathrm{s_{br1}}((i,i'),j,j') = \mathrm{s}^l_{\mathrm{br1}}(i,j) + \mathrm{s}^r_{\mathrm{br1}}(i',j') \text{ and } \mathrm{s_{br2}}(i,i',(j,j')) = \mathrm{s}^l_{\mathrm{br2}}(i,j) + \mathrm{s}^r_{\mathrm{br2}}(i',j'),$$

where $\mathrm{s}^l_{\mathrm{br1}}(i,j)$, $\mathrm{s}^r_{\mathrm{br1}}(i',j')$, $\mathrm{s}^l_{\mathrm{br2}}(i,j)$, and $\mathrm{s}^r_{\mathrm{br2}}(i',j')$ are the score contributions of the broken arc's left and right ends. For the similarity score, in the case of arc-complete alignments Equation 1 is equivalently defined by summing over the two ends of broken arcs independently. As a technicality, assume that the similarity score is defined in this way for arbitrary alignments, such that we score halve arc-breaks for non-arc-complete alignments.

[c]Note that Jiang *et al.* give only an approximation algorithm using their general score, since the problem is NP-complete for the general score.

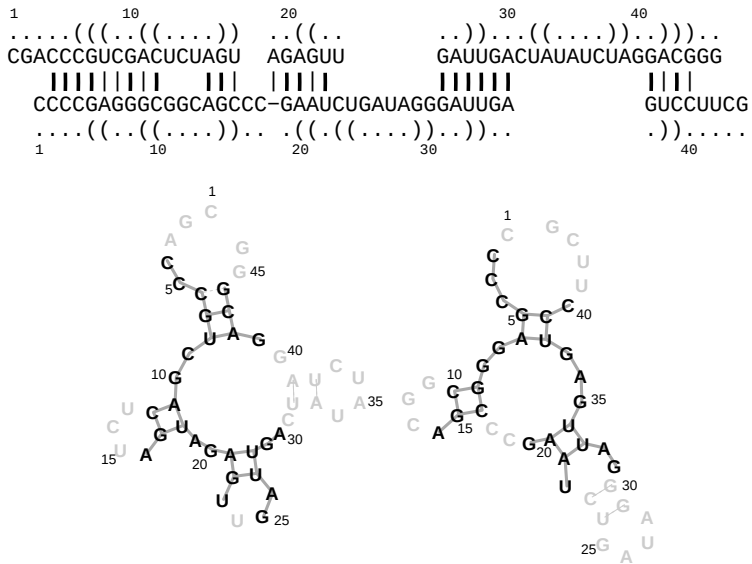

Fig. 2. A local alignment of two sequence-structures and their motif graphs in the alignment. These graphs, which represent the aligned, local parts, consist of only the dark nodes and dark grey edges. Nevertheless, for convenience we show the completed structures and unaligned arcs with light ink.

Note that distinguishing scores for left and right ends causes no additional work in the algorithm and is biologically justified, since RNA molecules are directed. However, for simplicity, we will not distinguish the two ends in the following and set for $l \in \{1,2\}$, $\mathrm{s}_{\mathrm{br}l}(i,j) = \mathrm{s}^{l}_{\mathrm{br}l}(i,j) = \mathrm{s}^{r}_{\mathrm{br}l}(i,j)$.

2.3. *Locality*

As mentioned in the introduction, our notion of locality implies that all bases in local motifs are connected. In this subsection, we will make this claim more precise by defining locality of a sequence-structure alignment and showing that locality is equivalent to the connectivity of corresponding motif graphs. Fig. 2 illustrates local alignment.

A *successor* of a base k is defined as an arc $(i,i') \in P$, such that (i,i') spans k (i.e. $i < k < i'$). The *immediate successor* of k which is its minimal successor w.r.t to $\prec_P$. For a range $[k..k']$, we denote (i,i') as a *successor* (resp. *immediate successor*) of $[k..k']$ if (i,j) is a *successor* (resp. *immediate successor*) of both k and k'. Note that, since our structures are nested, any range in a sequence-structure has at most one immediate successor, which is then determined unambigously.

Before defining lssa, we will introduce further terms. A denotes an alignment. We define an *exclusion of A in sequence* $l = 1,2$ as a range $[k..k']$, where $k \leq k'$, such that $[k..k'] \notin \pi_l(A)$ and $k-1, k'+1 \in \pi_l(A)$. Furthermore, we define P_l^A as the substructure of P_l that consists only of arcs that are aligned in A. For a node k in sequence l, we define *the immediate aligned successor for* k as the immediate successor of k in P_l^A . Similarly, we define the

immediate aligned successor for an exclusion $[k..k']$ of A to be the immediate aligned successor for both k and k'.

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

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

Now, local alignments are connected in the following sense. Let $\boldsymbol{\mathcal{S}} = (S, P)$ be a nested sequence-structure. Define $G_{\boldsymbol{\mathcal{S}}} = (V, E)$ as the graph where $V = \{i \mid 1 \leq i \leq |S|\}$, and $E = \{\{i, i+1\} | 1 \leq i < |S|\} \cup \{\{i, j\} \mid (i, j) \in P_l\}$. The *motif graph of a sequence-structure* $\boldsymbol{\mathcal{S}}_l$ $(l = 1, 2)$ *in an alignment* A is the subgraph G_l^A of $G_{\boldsymbol{\mathcal{S}}_l}$ consisting of only the bases and arcs aligned by A. Thus, the motif graph represents the aligned, local part of the sequence-structure. Formally, $G_l^A = (V_l, E_l)$ is the subgraph of $G_{\boldsymbol{\mathcal{S}}_l}$, where $V_l = \pi_l(A)$ and $E_l = \{\{i, i+1\} | i \in V_l, i+1 \in V_l\} \cup \{\{i, j\} \mid (i, j) \in P_l^A\}$.

Theorem 1. *Let A be an arc-complete alignment of $(\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{S}}_2)$, and G_l^A be the motif graphs of the sequence-structures $\boldsymbol{\mathcal{S}}_l$ $(l = 1, 2)$ in A. Then A is local if and only if the graphs G_l^A are connected.*

Proof sketch. For clarity, we are hiding some tedious technical details. The first direction, "local implies connected", is proven separately for both graphs by induction over the nested structures P_l^A in the $\prec_{P_l^A}$-ordering. Hence, we fix $l \in \{1, 2\}$. Then, one proofs inductively for every arc $(i, j) \in P_l^A$ the connectivity of the subgraphs $G_{i,j}$ and $G_{i,j}^{\text{loop}}$ of G_l^A, which are defined as follows. Let $(i, j) \in P_l^A$, then $G_{i,j}$ is the subgraph of G_l^A restricted to the nodes in $[i..j]$. An arc $(\hat{i}, \hat{j}) \in P_l^A$, where the arc (i, j) is the immediate (aligned) successor of the range $[\hat{i}..\hat{j}]$, is called *immediate (aligned) predecessor of* (i, j). $G_{i,j}^{\text{loop}}$ denotes the subgraph of $G_{i,j}$ that is restricted to the nodes in the outer multi-loop. Formally, $G_{i,j}^{loop}$ is generated from $G_{i,j}$ by removing the nodes in $[\hat{i}+1..\hat{j}-1]$, for each immediate aligned predecessor $(\hat{i}, \hat{j})$ of (i, j). An example of the subgraphs is given in Fig. 3.

Still, (i, j) denotes an arc in P_l^A. For the base case of the induction, there is no immediate aligned predecessor of (i, j). In consequence, $G_{i,j}^{\text{loop}}$ equals $G_{i,j}$. By the definition of locality, there is at most one exclusion with the immediate aligned successor (i, j). If there is such an exclusion $[k..k']$, then the graph consists of the nodes $[i..k-1]$ and $[k'+1..j]$. The nodes in each of the two subsets are connected, since they are consecutive, and the two subsets are connected by the arc (i, j).

For the induction step, for each immediate aligned predecessor $(\hat{i}, \hat{j})$ of (i, j), the graph $G_{\hat{i},\hat{j}}$ is connected by induction hypothesis. Then, by construction, $G_{i,j}$ is connected, if the corresponding loop-subgraph $G_{i,j}^{\text{loop}}$ is connected.

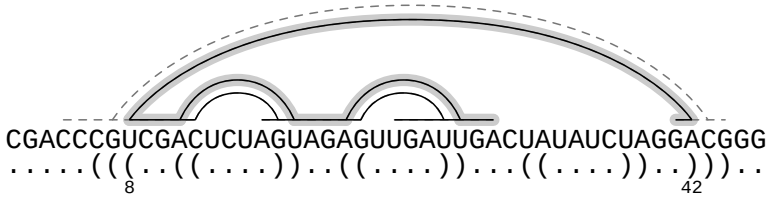

Fig. 3. The complete graph drawing represents the motif graph G_1^A for the alignment of Fig. 2. Recall that the motif graph contains only aligned bases and arcs. In particular there is no edge for the arc (7,43) and no nodes for the excluded bases CUAUAUCUAG. The subgraph $G_{8,42}$ of G_1^A is shown with black thin line, and the loop-subgraph $G_{8,42}^{\text{loop}}$ of G_1^A is drawn with gray thick line.

By definition of locality there is at most one exclusion $[k..k']$ with the immediate successor (i,j). We discuss the case that there is one. In order to show the connectivity of $G_{i,j}^{\text{loop}}$, we choose two nodes $s < t$ of $G_{i,j}^{\text{loop}}$ and show that they are connected (by case distinction). Note that the nodes smaller than k are connected in $G_{i,j}^{\text{loop}}$ since $[k..k']$ is the only exclusion with successor (i,j). The same holds for the nodes greater than k'. Therefore, the single remaining case is $s < k$ and $k' < t$. There, the connection of s and t is only due to the connection of i and j by the arc (i,j).

Instead of proving the second direction "connected implies local" directly, we prove its logically equivalent form "non local implies non connected". If the arc-complete alignment A is not local, then there are exclusions which contradict the second condition in the definition of local alignment. Either, there is one exclusion without a immediate aligned successor, or there are two exclusions sharing the same immediate successor. In both cases, we can identify sets of nodes in the motif graph that are isolated, since a connection contradicts the assumption that the structures are nested.

2.4. *Dynamic Programming Algorithm*

Here, we introduce a dynamic programming algorithm for the *lssa problem for sequence-structures* $\mathcal{S}_1 = (S_1, P_1)$ *and* $\mathcal{S}_2 = (S_2, P_2)$, where P_1 and P_2 are nested.

For applying dynamic programming to this problem, we develop several recursion equations, that recursively define the maximal similarity score of a local alignment of sequences S_1 and S_2. These recursion equations can be efficiently evaluated while filling matrices, i.e. materializing intermediate results. Finally, we obtain the actual optimal alignment by traceback from the matrices. We further argue in this subsection that the equations imply an $O(|S_1|^2 \cdot |S_2|^2)$ time and $O(|S_1| \cdot |S_2|)$ space lssa algorithm.

The recursion scheme presented below has been inspired by an algorithm developed by Jiang *et al.*[7] for finding global nested/crossing sequence-structure alignments. Whereas their algorithm is clearly presented by a single recursion equation, we have to introduce several interlocked recursion equations, in order to handle locality.

In the following, fix arcs $a_1 \in P_1$ and $a_2 \in P_2$. For sets I and J, we introduce $E(I,J)$

as an abbreviation for the set of edges $I \cup \{-\} \times J \cup \{-\}$. At the center of our system of recursion equations, we define $\mathrm{D}(a_1, a_2)$ as the maximal similarity score of a lssa $A \subseteq E([a_1^l..a_1^r], [a_2^l..a_2^r])$ of $\boldsymbol{\mathcal{S}}_1$ and $\boldsymbol{\mathcal{S}}_2$, where the two arcs a_1 and a_2 match. The scores are materialized in a matrix D and will be used to compute the maximal score of a lssa of $\boldsymbol{\mathcal{S}}_1$ and $\boldsymbol{\mathcal{S}}_2$ in the top-level computation step. For the recursive definition of an entry $\mathrm{D}(a_1, a_2)$ we will introduce four further recursion equations.

To compute a score $\mathrm{D}(a_1, a_2)$ we have to consider sub-sets of local alignments restricted to bases between the left and right ends of the two arcs a_1 and a_2. Since the arc-match between a_1 and a_2 justifies exclusions in the local alignments, those restrictions, which do not contain this arc-match, are not necessarily local alignments themselves. Furthermore, they need not to be arc-complete. We express this by the following definitions. For a structure P, we define the *restriction of P to a set I, write $P|I$,* as $\{(i,j) \in P \mid i,j \in I\}$. We call an alignment $A \subseteq E([i..i'], [j..j'])$ of $\boldsymbol{\mathcal{S}}_1$ and $\boldsymbol{\mathcal{S}}_2$ *sub-arc-complete* if it is arc-complete for $P_1|[i..i']$ and $P_2|[j..j']$. A *top-level exclusion in sequence l of alignment A* ($l = 1,2$) is an exclusion $[k..k']$ in sequence l of A, where neither k nor k' have a successor in P_l^A. A *local sub-alignment* $A \subseteq E([i..i'], [j..j'])$ of $\boldsymbol{\mathcal{S}}_1$ and $\boldsymbol{\mathcal{S}}_2$ is an alignment, satisfying the conditions of a local alignment except that in each sequence one top-level exclusion is allowed and the alignment is sub-arc-complete instead of arc-complete (compare to Definition 1).

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

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

(1) at most one exclusion in every sequence (arbitrary local sub-alignment) as ${}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}(i,j)$
(2) at most one exclusion in the first sequence as ${}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,j)$
(3) at most one exclusion in the second sequence as ${}^{\circ}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}(i,j)$
(4) no exclusions (true local alignment) as ${}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,j)$.

For applying dynamic programming, we will introduce four recursion equations, one for each state. Again, the intermediate results are materialized in matrices.

We define the scores in D using ${}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}$ by

$$\mathrm{D}(a_1, a_2) = {}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}(a_1^r - 1, a_2^r - 1) + \mathrm{s}_{\mathrm{arc}}(a_1, a_2).$$

Then, it remains to give the recursion equations for ${}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}$, ${}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}$, ${}^{\circ}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}$ and ${}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}$.

Since the cases, where no top-level exclusions are involved, are part of all recursion equations, we define a helper function $\textsc{NoEx}(\mathrm{M}^{a_1}_{a_2}, i, j)$ for any matrix $\mathrm{M}^{a_1}_{a_2} \in \{{}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}, {}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}, {}^{\circ}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}, {}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}\}$, where $(i,j) \in [a_1^l..a_1^r - 1] \times [a_2^l..a_2^r - 1]$.

$\textsc{NoEx}(\mathrm{M}^{a_1}_{a_2}, i, j)$ denotes the maximal similarity score of the alignments corresponding to the matrix entry $\mathrm{M}^{a_1}_{a_2}(i,j)$, which do not end (at the right) with an exclusion in either sequence. In consequence, we get

$${}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,j) = \textsc{NoEx}({}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}, i, j).$$

The maximal score $\text{NOEX}(\text{M}_{a_2}^{a_1}, i, j)$ can stem from four classes of alignments. In the first case, the alignments match $S_1[i]$ and $S_2[j]$. Then, the score is computed as the sum of $\text{M}_{a_2}^{a_1}(i-1, j-1)$, the similarity of the matched bases, and possibly the penalty (negative similarity) for breaking the arcs, that are incident to i or j. In the next two cases, the alignment ends in a gap in the first sequence (resp. the second sequence). The score is the sum of $\text{M}_{a_2}^{a_1}(i-1, j)$ (resp. $\text{M}_{a_2}^{a_1}(i-1, j)$), the gap dissimilarity, and possibly a term for arc-breaking. In the last case, an alignment ends in an arc-match. There, we add the maximal similarity of alignments left of the arc match to the maximal similarity for alignments that are framed by the arc-match given in D.

In the following definition of NOEX, we use the helper functions inc_1 and inc_2 for a compact notation. As defined in our preliminaries $inq(i) = 1$ if there is an incident arc to base i in the structure P_l and $inc_l(i) = 0$ otherwise (for $l = 1, 2$). Multiplying one term by $\text{inc}_l(i)$ and another term by $1 - \text{inc}_l(i)$ supports the distinction of cases.

$$\text{NOEX}(\text{M}_{a_2}^{a_1}, i, j) = \max \begin{cases} \text{M}_{a_2}^{a_1}(i-1, j-1) + (1-\text{inc}_1(i))(1-\text{inc}_2(j))\text{s}_\text{b}(i, j) \\ \quad + \text{inc}_1(i)\text{s}_{\text{br}1}(i, j) + \text{inc}_2(j)\text{s}_{\text{br}2}(i, j) \\ \text{M}_{a_2}^{a_1}(i-1, j) + (1-\text{inc}_1(i))\text{s}_\text{b}(i, -) + \text{inc}_1(i)\text{s}_{\text{br}1}(i, -) \\ \text{M}_{a_2}^{a_1}(i, j-1) + (1-\text{inc}_2(j))\text{s}_\text{b}(-, j) + \text{inc}_2(j)\text{s}_{\text{br}2}(-, j) \\ \text{M}_{a_2}^{a_1}(i'-1, j'-1) + \text{D}((i', i), (j', j)) \\ \quad \text{iff } (i', i) \in P_1 \text{ and } (j', j) \in P_2. \end{cases}$$

For the initial cases, one of the subsequences has zero length, i.e. if $i = a_1^l$ or $j = a_2^l$. Since no top-level exclusions are handled here, we get

$$\begin{aligned} \text{NOEX}(\text{M}_{a_2}^{a_1}, a_1^l, a_2^l) &= 0 \\ \text{NOEX}(\text{M}_{a_2}^{a_1}, i, a_2^l) &= \text{M}_{a_2}^{a_1}(i-1, a_2^l) + (1-\text{inc}_1(i))\text{s}_\text{b}(i, -) + \text{inc}_1(i)\text{s}_{\text{br}1}(i, -) \\ \text{NOEX}(\text{M}_{a_2}^{a_1}, a_1^l, j) &= \text{M}_{a_2}^{a_1}(a_1^l, j-1) + (1-\text{inc}_2(j))\text{s}_\text{b}(-, j) + \text{inc}_2(j)\text{s}_{\text{br}2}(-, j). \end{aligned}$$

Now we define the recursion cases for the remaining matrices ${}_{\circ}^{\text{x}}\text{M}_{a_2}^{a_1}$, ${}_{\text{x}}^{\circ}\text{M}_{a_2}^{a_1}$, and ${}_{\text{x}}^{\text{x}}\text{M}_{a_2}^{a_1}$. Beside the recursion cases, where no new exclusion is inserted, we need additional cases to introduce new top-level exclusions, which are arc-complete. (Note that one proves by contradiction that every exclusion of a local alignment A is itself arc-complete.). The additional cases reflect the insertion of an exclusion that starts with i in the first sequence (resp. j in the second sequences), and extends to left until some $k < i$ (resp. $k < j$) such that the exclusion $[k+1..i]$ (resp. $[k+1..j]$) is arc-complete. We need to examine all such values of k[d]. This results for $(i, j) \in [a_1^l..a_1^r - 1] \times [a_2^l..a_2^r - 1]$ in the recursion equations

$${}_{\circ}^{\circ}\text{M}_{a_2}^{a_1}(i, j) = \text{NOEX}({}_{\circ}^{\circ}\text{M}_{a_2}^{a_1}, i, j),$$

[d]Since we already run over these values of k, here it is possible to impose further restrictions on the admissible exclusions, e.g. it seems reasonable to limit the minimal length of exclusions, which can be done without additional work in the algorithm.

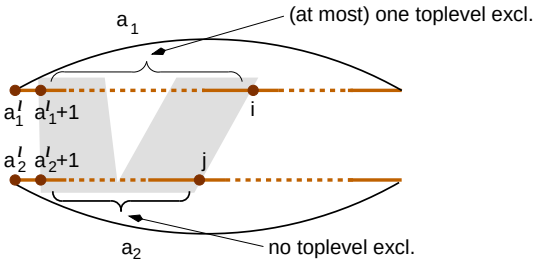

Fig. 4. Local sub-alignment considered in ${}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,j)$. The areas of alignment edges are marked grey.

$$
{}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,j) = \max \begin{cases} \textsc{NoEx}({}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}, i, j) \\ \max\limits_{a_1^l \le k < i, \mathrm{ac}_1([k+1..i])} {}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}(k,j), \end{cases}
$$

$$
{}^{\circ}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}(i,j) = \max \begin{cases} \textsc{NoEx}({}^{\circ}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}, i, j) \\ \max\limits_{a_2^l \le k < j, \mathrm{ac}_2([k+1..j])} {}^{\circ}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,k), \end{cases}
$$

$$
\text{and } {}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}(i,j) = \max \begin{cases} \textsc{NoEx}({}^{\mathrm{x}}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}, i, j) \\ \max\limits_{a_1^l \le k < i, \mathrm{ac}_1([k+1..i])} {}^{\circ}_{\mathrm{x}}\mathrm{M}^{a_1}_{a_2}(k,j) \\ \max\limits_{a_2^l \le k < j, \mathrm{ac}_2([k+1..j])} {}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,k). \end{cases}
$$

Note that for $i = a_1^l$ (resp. $j = a_2^l$) the maximizations run over empty ranges. We define such a maximum over an empty range as $-\infty$. For example, this sets the entries (a_1^l, a_2^l) of the four matrices to 0. Fig. 4 shows a local sub-alignment handled by equation ${}^{\mathrm{x}}_{\circ}\mathrm{M}^{a_1}_{a_2}(i,j)$.

Finally, we use the our helper function NOEX (and thus implicitly the table D) for defining recursion equations for the complete alignment, i.e. for the top-level of our alignment. Here, we look for the best scoring alignment of a pair of subsequences of S_1 and S_2, which is similar to local sequence alignment. In the case of local sequence alignment, this is done by dropping prefix and suffix alignments with negative scores. However, in our case the aligned subsequences have to be arc-complete. In consequence, one cannot simply drop arbitrary prefixes, as it is done in local sequence alignment.

However, we can avoid searching through all possible starting and ending points of the subsequences. Fortunately, searching over all $(a_1^l + 1, a_2^l + 1)$ as starting positions, where a_1 (resp. a_2) is an arc in P_1 (resp. P_2) or spans the whole sequence, turns out to suffice. If we additionally allow the dropping of negatively scored arc-complete prefix alignments in the recursion equation, we can take account for every pair of arc-complete subsequences.

For defining the top-level recursion, fix start indices i_0 and j_0. Choose i_2 and j_2 to be maximal such that $\mathrm{ac}_1([i_0..i_2])$ and $\mathrm{ac}_2([j_0..j_2])$. We define $\mathrm{T}(i,j)$ for $(i,j) \in [i_0 - 1..i_2] \times$

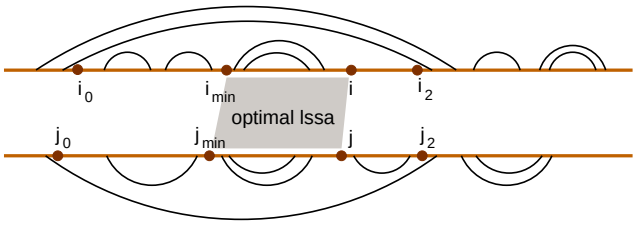

Fig. 5. Example for the last step ("top-level") of the local alignment. The figure illustrates the meaning of the indices that are used in the description. The region of the optimal lssa that is scored by $T(i,j)$ is highlighted.

$[j_0-1..j_2]$ recursively by

$$\mathrm{T}(i,j) = \max\left(\textsc{NoEx}(\mathrm{T},i,j), \left\{\begin{array}{ll} 0 & \text{if } \mathrm{ac}_1([i_0..i]) \wedge \mathrm{ac}_2([j_0..j]) \\ -\infty & \text{otherwise} \end{array}\right\}\right).$$

Note that by limiting the score of arc-complete prefix alignments to be at least zero, we accomplish the effect of dropping negatively scored prefix alignments. For $\mathrm{ac}_1([i_0..i]) \wedge \mathrm{ac}_2([j_0..j])$, $T(i,j)$ yields the score of the optimal lssa $A \subseteq E([i_0..i],[j_0..j])$, where i and j are aligned and for the minimal positions $i_{\min}$ and $j_{\min}$ that are aligned by A, holds $\mathrm{ac}_1([i_0..i_{\min}-1])$ and $\mathrm{ac}_2([j_0..j_{\min}-1])$. Fig. 5 illustrates the top-level alignment.

Finally, to get the maximal score we search through all $i_0 = a_1^l + 1$ and $j_0 = a_2^l + 1$ for arc-pairs $(a_1,a_2) \in P_1 \times P_2$, and determine the maximal entry (i_1,j_1) in the matrices T, where $\mathrm{ac}_1([i_0..i_1])$ and $\mathrm{ac}_2([j_0..j_1])$.

Theorem 2. *For a similarity score* SIMSCORE *and sequence-structures* $\boldsymbol{\mathcal{S}}_1 = (S_1,P_1)$ *and* $\boldsymbol{\mathcal{S}}_2 = (S_2,P_2)$ *with nested structures* P_1 *and* P_2*, there is a* $O(|S_1|^2 \cdot |S_2|^2 \cdot \max(|S_1|,|S_2|))$ *time and* $O(|S_1| \cdot |S_2|)$ *space algorithm for the lssa problem.*

Proof sketch. The existence of an algorithm has been shown already. Albeit a rigorous proof of the algorithm's correctness has been omitted. Since the scores in the matrices D, $\mathrm{M}_{a_2}^{a_1}$, and T are explicitly defined as maximal scores of certain classes of alignments, the correctness is proven by showing the equivalence of these definitions to the recursion equations. This is shown inductively.

It remains to discuss the algorithm's complexity. The algorithm computes and stores the $O(|S_1| \cdot |S_2|)$ many entries of the matrix D. For each entry of D, we compute the $O(|S_1| \cdot |S_2|)$ many entries of the matrices $\mathrm{M}_{a_2}^{a_1}$. For computing each of the entries in D, we compute separate matrices $\mathrm{M}_{a_2}^{a_1}$. However, since there is no dependency between the matrices $\mathrm{M}_{a_2}^{a_1}$ for different arc-pairs, it suffices to store the matrix D as well as the matrix T and only one instance of the matrices $\mathrm{M}_{a_2}^{a_1}$ at one time. Hence, we need only $O(|S_1| \cdot |S_2|)$ space.

Regarding time complexity, in the computation of the matrices $\mathrm{M}_{a_2}^{a_1}$, we maximize over all values of k in a certain range, where its size is limited by the sequence length. The complexity of computing D is dictated by the total number of steps we perform in these maximizations, since the other cases of our recursion equations are evaluated in constant time.

The worst case time complexity of the algorithm is thus $O(|S_1|^2 \cdot |S_2|^2 \cdot \max(|S_1|, |S_2|))$, since the top-level computation has a time complexity of only $O(|S_1|^2 \cdot |S_2|^2)$ and finally the traceback step, where we recompute the matrices $\mathrm{M}^{a_1}_{a_2}$, does not increase time or space complexity.

3. Results

An implementation of the introduced algorithm written in C++ is available on the webpage *http://www.bio.inf.uni-jena.de*. We have aligned RNase P from two different organisms *Ralstonia eutrophus* and *Streptomyces bikiniensis*. Whereas the algorithm allows for fine-tuning of parameters, here the scores are chosen ad-hoc.[e] The example is computed in about 75 seconds on a Intel Pentium 4 running at 2.4GHz, while the alignment of Fig. 2 is computed in a few milliseconds. Jiang *et al.*'s implementation needs approximately the same time to produce a global alignment. The resulting local alignment is shown in Fig. 6. It shows sequences and structures as well as the aligned bases. Only the bases that are connected by | are aligned. Fig. 8 shows the secondary structures of the two RNA molecules, where the parts included and excluded by the alignment of Fig. 6 are distinguished.

The sequences and structures are taken from Brown's RNase P Database[16]. The reader may compare our example to a corresponding example given by Jiang *et al.*[7], where the two RNase P molecules are aligned globally.[f] As a drawback of giving this opportunity for a comparison, the two molecules, which were originally chosen for global alignment, have small dissimilar parts. In consequence, the exclusions are small and there are no unaligned prefixes or suffixes.

As a further test-case, we used our pairwise lssa algorithm to produce input for T-coffee[17]. The result of aligning SECIS elements of M.janaschii from Wilting *et al.*[2] is shown in Fig. 7.

Acknowledgment

We thank the anonymous referees, who helped to improve the quality of the paper. Furthermore, we thank Sven Siebert for the many discussions on global and local sequence-structure alignment and his help with the multiple alignment example.

References

1. J. Cousin. Breakthrough of the year: Small RNAs make big splash. *Science*, 298:2296–97, 2002.
2. R. Wilting, S. Schorling, B. C. Persson, and A. Böck. Selenoprotein synthesis in archaea: Identification of an mRNA element of Methanococcus jannaschii probably directing selenocysteine insertion. *J. Mol. Biol.*, 266(4):637–41, 1997.
3. SJ Klug and M Famulok. All you wanted to know about selex. *Mol. Biol. Rep.*, 20(2):97–107, 1994.

[e] We are aware that, in general, the quality of predicted alignments depends strongly on the actual parameters for the similarity score. Unfortunately, there are no theoretical foundations to estimate such parameters systematically.
[f] Jiang et al. refer to the recently renamed organism *R. eutrophus* by *Alcaligenes eutrophus*

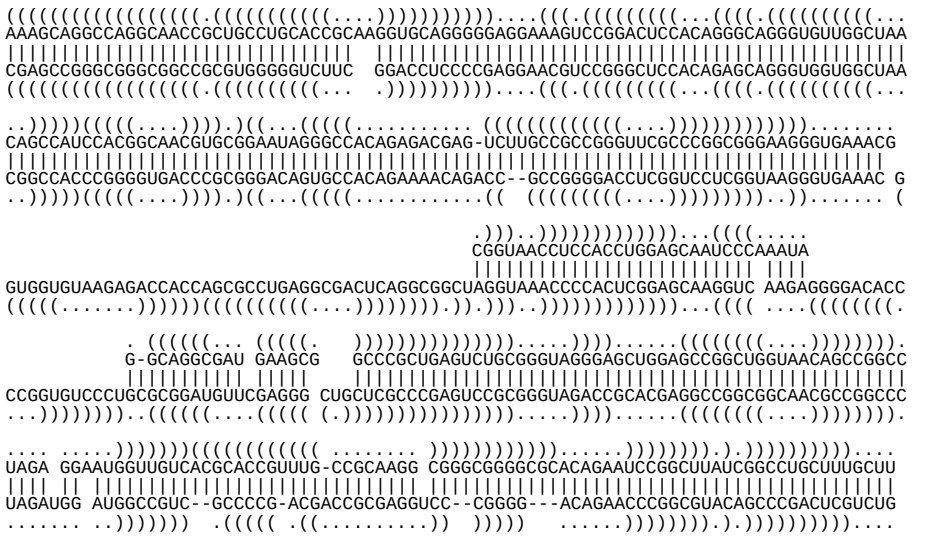

Fig. 6. Local sequence-structure alignment of RNase P of *R.eutrophus* and *S.bikiniensis*. The sequences and structures are taken from the RNase P Database by Brown[16]. Only the bases that are shown connected by | are aligned. The algorithm correctly identifies and excludes substructures that occur only in one of the RNA molecules.

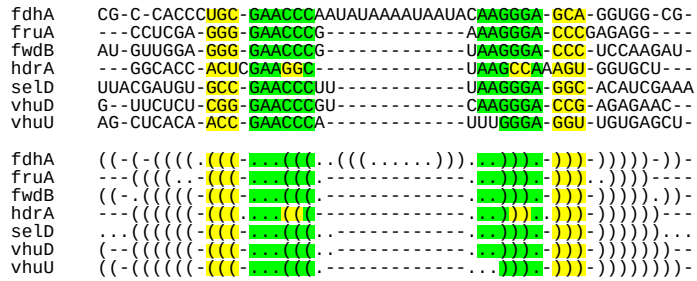

Fig. 7. Alignment of SECIS elements from Methanococcus janaschii using lssa. The bases, which supposedly belong to the common motif are colored, where we further mark the bases that occur in both sequences identically.

4. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215(3):403–10, 1990.
5. M. Zuker. Prediction of RNA secondary structure by energy minimization. *Meth. Mol. Biol.*, 25:267–94, 1994.
6. S. Wuchty, W. Fontana, I. L. Hofacker, and P. Schuster. Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers*, 49(2):145–65, 1999.
7. T. Jiang, G. Lin, B. Ma, and K. Zhang. A general edit distance between RNA structures. *J. Comput. Biol.*, 9(2):371–88, 2002.
8. H. P. Lenhof, K. Reinert, and M. Vingron. A polyhedral approach to RNA sequence structure alignment. *J. Comput. Biol.*, 5(3):517–30, 1998.
9. S. R. Eddy. A memory-effi cient dynamic programming algorithm for optimal alignment of a

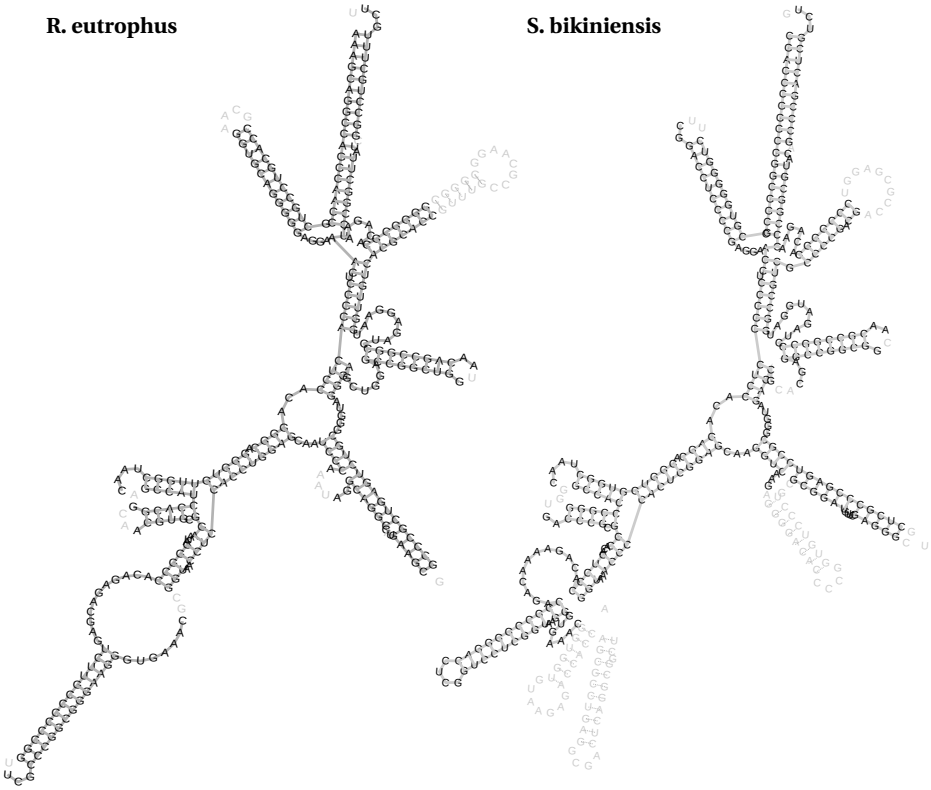

Fig. 8. Secondary structures of RNase P molecules. We show the parts of the structures that are not aligned by our algorithm only lightly. Notably, in the presented form, the algorithm produces small exclusions, which can be prevented easily, if wanted. Please also see Footnote d on page 11 on this issue.

sequence to an RNA secondary structure. *BMC Bioinformatics*, 3(1):18, 2002.

10. J. Gorodkin, S. L. Stricklin, and G. D. Stormo. Discovering common stem-loop motifs in unaligned RNA sequences. *Nucleic Acids Res.*, 29(10):2135–44, 2001.
11. M. Höchsmann, T. Töller, R. Giegerich, and S. Kurtz. Local similarity in rna secondary structures. In *Proc. of Comput. Sys. Bioinf. (CSB 2003)*, 2003.
12. T. Jiang, J. Wang, and K. Zhang. Alignment of trees - an alternative to tree edit. *Theor. Comput. Sci.*, 143(1):137–148, 1995.
13. M. Gerstein and M. Levitt. Comprehensive assessment of automatic structural alignment against a manual standard, the scop classifi cation of proteins. *Prot. Sci.*, 7(2):445–56, 1998.
14. G. Lancia, R. Carr, B. Walenz, and S. Istrail. 101 optimal PDB structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. In *Proc. of* 5^{th} *Int. Conf. on Comput. Mol. Biol. (RECOMB 2001)*, 2001.
15. J. Xu, M. Li, D. Kim, and Y. Xu. RAPTOR: Optimal protein threading by linear programming. *J. Bioinf. Comput. Biol.*, 2003.
16. J. W. Brown. The ribonuclease P database. *Nucleic Acids Res.*, 27(1):314, 1999.
17. C. Notredame, D. G. Higgins, and J. Heringa. T-coffee: A novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.*, 302(1):205–17, 2000.