BIOINFORMATICS ORIGINAL PAPER Vol. 25 no. 16 2009, pages 2095–2102
doi:10.1093/bioinformatics/btp065

Sequence analysis

Lightweight comparison of RNAs based on exact sequence–structure matches

Steffen Heyne, Sebastian Will, Michael Beckstette and Rolf Backofen*

Bioinformatics Group, Albert-Ludwigs-University Freiburg, Georges-Koehler-Allee 106, Freiburg, D-79110, Germany

Received and revised on October 31, 2008; accepted on January 26, 2009

Advance Access publication February 2, 2009

Associate Editor: Trey Ideker

ABSTRACT

Motivation: Specific functions of ribonucleic acid (RNA) molecules are often associated with different motifs in the RNA structure. The key feature that forms such an RNA motif is the combination of sequence and structure properties. In this article, we introduce a new RNA sequence–structure comparison method which maintains exact matching substructures. Existing common substructures are treated as whole unit while variability is allowed between such structural motifs.

Based on a fast detectable set of overlapping and crossing substructure matches for two nested RNA secondary structures, our method `ExpaRNA` (exact pattern of alignment of RNA) computes the longest collinear sequence of substructures common to two RNAs in $O(H \cdot nm)$ time and $O(nm)$ space, where $H \ll n \cdot m$ for real RNA structures. Applied to different RNAs, our method correctly identifies sequence–structure similarities between two RNAs.

Results: We have compared `ExpaRNA` with two other alignment methods that work with given RNA structures, namely `RNAforester` and `RNA_align`. The results are in good agreement, but can be obtained in a fraction of running time, in particular for larger RNAs. We have also used `ExpaRNA` to speed up state-of-the-art Sankoff-style alignment tools like `LocARNA`, and observe a tradeoff between quality and speed. However, we get a speedup of 4.25 even in the highest quality setting, where the quality of the produced alignment is comparable to that of `LocARNA` alone.

Availability: The presented algorithm is implemented in the program `ExpaRNA`, which is available from our website (http://www.bioinf.uni-freiburg.de/Software).

Contact: {exparna,backofen}@informatik.uni-freiburg.de

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Ribonucleic acids (RNAs) are associated with a large range of important cellular functions in living organisms. Moreover, recent findings show that RNAs can perform regulatory functions formerly assigned only to proteins. Likewise to proteins, these functions are often associated with evolutionary conserved motifs that contain specific sequence and structure properties. Examples for such regulatory RNA elements, whose functions are mediated by sequence–structure motifs are selenocysteine insertion sequence (SECIS) elements (Huttenhofer *et al.*, 1996) (see Fig. 1 for an example), iron-responsive elements (IREs) (Hentze and Kuhn, 1996), different riboswitches (Serganov and Patel, 2007) or internal ribosomal entry sites (IRESs) (Martineau *et al.*, 2004). Therefore, the detection of similar structural motifs in different RNAs is an important aspect for function determination and should be considered in pairwise RNA comparison methods. Although this problem is addressed in sequence–structure alignment methods, these approaches are often very time-consuming and do not necessarily preserve functionally important common substructures in the alignment (Jiang *et al.*, 1995, 2002).

In this article, we propose a new lightweight, motif-based method for the pairwise comparison of RNAs. Instead of computing a full sequence–structure alignment, our approach efficiently computes a significant arrangement of sequence–structure motifs, common to two RNAs. For the sake of algorithmic complexity and applicability in practice, we neglect higher order interactions like pseudoknots. This allows to describe sequence–structure motifs with nested RNA secondary structures, as shown in Figure 1.

Our `ExpaRNA` (exact pattern of alignment of RNA) method uses as a pre-processing step a fast $O(nm)$ time and space algorithm from Backofen and Siebert (2007) for the identification of isolated common substructures for the two given RNAs of lengths n and m with nested secondary structures. More precisely, this method identifies the complete, but overlapping set of exact common substructures. Our approach makes use of these common substructures and computes the longest collinear, non-overlapping sequence of substructures common to two RNAs in $O(H \cdot nm)$ time and $O(nm)$ space, where $H \ll n \cdot m$ for real RNA structures. Herein after, we call this the LONGEST COMMON SUBSEQUENCE OF EXACT PATTERN MATCHINGS problem (LCS-EPM).

The LCS-EPM requires known or predicted structure. We have compared our approach with two other alignment methods that work with given RNA structures, namely `RNAforester` and `RNA_align`. The results are in good agreement, but can be obtained in a fraction of running time, in particular for larger RNAs.

Since in many practical applications, there is no known structure, and structure prediction would lead to wrong results, we have also setup a pipeline that combines `ExpaRNA` with a state-of-the-art Sankoff-style algorithm for simultaneous alignment and folding (Sankoff, 1985). Albeit Sankoff-like approaches are currently the gold standard for RNA alignment, it has the drawback

*To whom correspondence should be addressed.

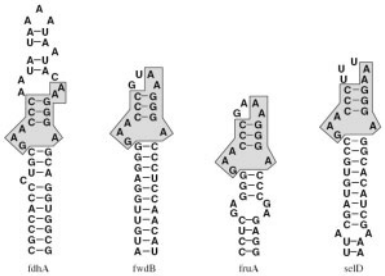

Fig. 1. Putative SECIS elements in non-coding regions of *Methanococcus jannaschii* according to Wilting *et al.* (1997). The indicated substructure represents a common substructure, i.e. a local motif.

of a high computational complexity. Basically, we predict a longest common subsequence of exact pattern first, and then use LocARNA (Will *et al.*, 2007) to fill the unaligned space between the exact pattern matchings. This amounts to calculate a constraint alignment by LocARNA, which restricts the search space and thus speeds up LocARNA. Moreover, the speedup increases with the extent of information calculated by ExpaRNA. However, this normally implies that the quality is decreased. Hence, there is a trade-off between the speedup resulting from this combined pipeline, and the quality of the produced alignment. However, we get a speedup of 4.25 even in the highest quality setting, where the quality of the produced alignment is comparable to that of LocARNA alone. In application scenarios where optimal quality is not strictly required, we obtain a speedup up to 8.25. Note that this pipeline could also be used in combination with other Sankoff-like tools that are in principle able to profit from alignment constraints, e.g. Dynalign, PMComp and FoldalignM (Hofacker *et al.*, 2004; Mathews and Turner, 2002; Torarinsson *et al.*, 2007).

Related work: existing approaches addressing the sequence–structure comparison problem for RNA molecules can be distinguished by the given structural information and their representation. The standard alignment-based comparison approach employs the computation of edit distances between given RNA secondary structures (Bafna *et al.*, 1995; Jiang *et al.*, 2002). In (Evans, 1999) the author introduced the problem of finding the longest arc-preserving common subsequence (LAPCS). However, even for two *nested* RNA secondary structures, both problems remain NP-hard (Blin *et al.*, 2003; Lin *et al.*, 2002). With some restrictions to the scoring scheme, the time complexity for determination of the edit distance can be lowered to polynomial time (Jiang *et al.*, 2002).

If the nested secondary structure is represented as a tree, comparison methods exist for the edit distance between two ordered labeled trees (Zhang and Shasha, 1989) as well as for the alignment of trees (Jiang *et al.*, 1995). An improved version of the tree alignment method with extension to global and local forest alignments is given in Höchsmann *et al.* (2003) and implemented in the program RNAforester. The MiGaL (Allali and Sagot, 2005) approach extends the tree edit distance model by the two new tree edit operations and is especially efficient due to its usage of different abstraction layers.

The article is organized as follows. In Section 2, we describe the way in which exact common substructures can be used for pairwise sequence–structure comparison. In addition, we explain how sequence–structure alignment methods can profit from anchor constraints. Sections 3 and 4 present the results for two applications of our tool ExpaRNA.

2 METHODS

RNA is a macro molecule described formally by a pair $\mathcal{R}=(S,B)$ of a primary structure S and a secondary structure B. A *primary structure* S is a sequence of nucleotides $S=s_1s_2\ldots s_n$ over the alphabet $\{A,C,G,U\}$. With $|S|$ we denote the length of sequence S. $S[i]$ indicates the nucleotide at position i in sequence S. With $S[i...j]$ we define the substring of S starting at position i until j for $1\leq i<j\leq|S|$. A *secondary structure* B is a set of base pairs $B=\{(i,i')\mid 1\leq i<i'\leq|S|\}$ over S, where each base takes part in at most one base pair. A secondary structure B is called *crossing* if there are two pairs $(i,i'),(j,j')\in B$ with $i<j<i'<j'$. Otherwise it is called *non-crossing* or *nested*.

For the definition of local RNA motifs, we represent an RNA $\mathcal{R}=(S,B)$ as undirected labeled graph $G=(V,E)$, called the *structure graph of* $\mathcal{R}$. Its set of vertices V is the set of positions in S, i.e. $V=\{1,\ldots,|S|\}$. Its set of edges E comprises all backbone bonds and all base pairs, i.e. $E=\{(i,i+1)\mid 1\leq i<|S|\}\cup B$. An *RNA pattern in* $\mathcal{R}$ is a set of positions $\mathcal{P}\subseteq\{1,\ldots,|S|\}$, such that the *pattern graph for* $\mathcal{P}$ *in* G, defined as the subgraph $G'=(V',E')$ of G, where $V'=\mathcal{P}$ and $E'=\{(i,i')\in E\mid i\in\mathcal{P}\text{ and }i'\in\mathcal{P}\}$, is connected. By this definition, an RNA pattern corresponds to a local motif, i.e. a substructure consisting of neighbored nucleotides according to a neighborhood that is induced by the backbone bonds and base pairs within a fixed secondary structure (cf. Fig. 1).

2.1 Exact pattern matchings of two RNAs

In the following, we consider two fixed, non-crossing RNAs $\mathcal{R}_1=(S_1,B_1)$ and $\mathcal{R}_2=(S_2,B_2)$. Their corresponding structure graphs are $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$, respectively. We will define an exact pattern matching as a special *ordered matching of* V_1 *and* V_2, i.e. as a set $\mathcal{M}\subseteq V_1\times V_2$, where for all $(p,q),(p',q')\in\mathcal{M}$ it holds that $p<p'$ implies $q<q'$ and $p=p'$ iff $q=q'$.

According to an ordered matching $\mathcal{M}$ of V_1 and V_2, we merge the graphs G_1 and G_2 into a *matching graph* $G_{\mathcal{M}}=(\mathcal{M},E_{\mathcal{M}})$, where $E_{\mathcal{M}}=\{((p,q),(p',q'))\in\mathcal{M}\times\mathcal{M}\mid(p,p')\in E_1\text{ and }(q,q')\in E_2\}$. A pair $(p,q)\in\mathcal{M}$ is called *admissible* if it satisfies the following conditions: (i) $S_1[p]=S_2[q]$ and (ii) $\mathsf{STRUCT}_1(p)=\mathsf{STRUCT}_2(q)$. Here, function $\mathsf{STRUCT}_i(j)$ yields one of the three possible structural types for a nucleotide at position j in structure i: *single stranded*, *left paired* or *right paired*. Furthermore, exact pattern matchings need to preserve all base pairs. A matching $\mathcal{M}$ satisfies this iff $\forall(p,q),(p',q')\in\mathcal{M}:(p,p')\in B_1\Leftrightarrow(q,q')\in B_2$. Then, an *exact pattern matching* $\mathcal{PM}$ is an ordered matching where $G_{\mathcal{PM}}$ is connected, all $(p,q)\in\mathcal{PM}$ are admissible and all base pairs are preserved.

Hence, an exact pattern matching $\mathcal{PM}$ describes the matching between sets of positions in the two RNAs $\mathcal{R}_1$ and $\mathcal{R}_2$, namely the projections $\pi_1\mathcal{PM}=\{p\mid(p,q)\in\mathcal{PM}\}$ and $\pi_2\mathcal{PM}=\{q\mid(p,q)\in\mathcal{PM}\}$. Note that $\pi_1\mathcal{PM}$ and $\pi_2\mathcal{PM}$ are patterns in $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively, i.e. in particular they correspond to the connected pattern graphs G_1^p and G_2^p. Note further, although we require that an exact pattern matching $\mathcal{PM}$ is an isomorphism on base pairs, $\mathcal{PM}$ does not necessarily describe an isomorphism on backbone edges in the pattern graphs G_1^p and G_2^p, since for $(p,q),(p',q')\in\mathcal{PM}$ where p and p' form an edge in G_1^p, q and q' do not necessarily form an edge in G_2^p. For details and proofs we refer to Backofen and Siebert (2007).

For our algorithm, we utilize only *maximal* exact pattern matchings, i.e. $\forall\mathcal{PM}':\mathcal{PM}\subseteq\mathcal{PM}'\Rightarrow\mathcal{PM}'=\mathcal{PM}$. In the following, we abbreviate the term maximal exact matching pattern by EPM. Similar to the minimal word size as e.g. used in BLAST (Altschul *et al.*, 1997), it is reasonable to consider a minimal size γ for EPMs. Hence, the set of all maximal exact pattern matchings $\mathcal{E}$ over two RNAs $\mathcal{R}_1$ and $\mathcal{R}_2$ is defined as

$$\mathbf{E}_{\gamma}^{1,2}=\{\,\mathcal{E}\mid\mathcal{E}\text{ is EPM}\wedge|\mathcal{E}|\geq\gamma\,\}.$$

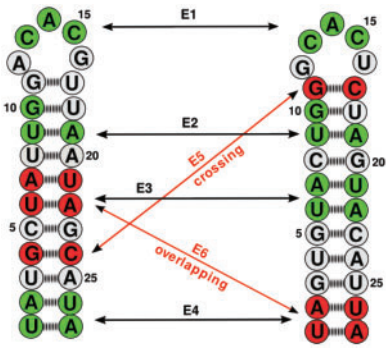

Fig. 2. A possible set $\mathbf{E}_{\gamma}^{1,2}$ for two RNAs $\mathcal{R}_1$, $\mathcal{R}_2$. The set $\{\mathcal{E}_1,\mathcal{E}_2,\mathcal{E}_3,\mathcal{E}_4\}$ can be used for a comparison, whereas $\{\mathcal{E}_5,\mathcal{E}_6\}$ should be excluded. $\mathcal{E}_5$ is crossing $\mathcal{E}_2$ and $\mathcal{E}_3$, whereas $\mathcal{E}_6$ is overlapping with $\mathcal{E}_3$ in $\mathcal{R}_1$ and with $\mathcal{E}_4$ in $\mathcal{R}_2$. Note that not all possible EPMs are indicated.

Note that each EPM is an arc-preserving common (but not longest common) subsequence as defined in Evans (1999) for the LAPCS problem. Since EPMs have in addition the above described properties, the detection of all EPMs is a computationally light problem, compared to LAPCS, which is NP-complete even for nested sequences (Blin *et al.*, 2003). Using the dynamic programming approach described in Backofen and Siebert (2007), the set of all EPMs can be found in $O(nm)$ time and $O(nm)$ space, making this approach applicable for fast sequence–structure comparisons. Now recall that each EPM is maximal. This implies that any two exact pattern matchings are disjoint and therefore a pair $(p,q)\in\mathcal{E}\in\mathbf{E}_{\gamma}^{1,2}$ is unique in $\mathbf{E}_{\gamma}^{1,2}$ and part of at most one EPM. The number of EPMs contained in $\mathbf{E}_{\gamma}^{1,2}$ is bounded by $n\cdot m$, with $n=|S_1|$ and $m=|S_2|$.

$\mathbf{E}_{\gamma}^{1,2}$ can be seen as a 'library' of all common motifs between two RNAs that can be utilized for a pairwise comparison method. Thus, the main idea of our approach will be to take a subset EPMs from $\mathbf{E}_{\gamma}^{1,2}$ that in combination will cover a large portion of both RNAs. The EPMs in $\mathbf{E}_{\gamma}^{1,2}$ differ in their size and shape as well as in their structural positions in both RNAs. Simply selecting two or several of these substructures for combination would probably lead to overlapping or crossing structures (Fig. 2). Hence, the set of all EPMs is not a solution for the LAPCS problem since the combination of several EPMs is not necessarily arc-preserving. Clearly, a meaningful subset of common substructures excludes overlapping and crossing patterns. This guarantees that the backbone order of matched nucleotides as well as base pairs of the given RNAs are preserved. Compatible EPMs are called non-crossing. Formally, two EPMs $\mathcal{E}_1$ and $\mathcal{E}_2$ are *non-crossing* if $\mathcal{E}_1\cup\mathcal{E}_2$ is an ordered matching. Figure 2 shows an example of a possible set $\mathbf{E}_{\gamma}^{1,2}$. A 'good' subset to describe the similarity between the two RNAs would probably exclude the EPMs indicated in red.

2.2 Combining EPMs for comparing RNAs: problem definition and algorithm overview

The formulation of LCS-EPM is motivated by the fact that similar RNAs with fixed secondary structures share identical structural elements in a similar arrangement. Examples are shown in our result section for the comparison of thermodynamically folded as well as experimentally verified secondary structures. The knowledge of such a 'common core' of identical substructures in two RNAs is interesting for different tasks.

For our global approach, we are interested in a *maximal* possible arrangement of substructures shared by two RNAs. If the motifs are given in the form of exact pattern matchings, we call this the LCS-EPM problem. Basically, we search for a maximal combination of EPMs that form a common subsequence. Note that albeit the problem shares some similarity with LAPCS, it is restricted in such a way that an efficient solution is possible.

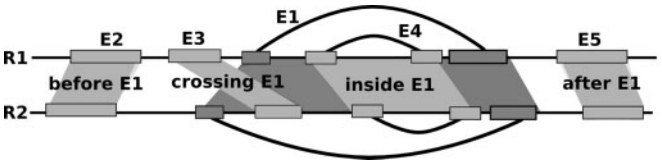

Fig. 3. Ordering of exact pattern matchings relative to EPM $\mathcal{E}_1$ (indicated in dark gray). The cases *before, inside* and *after* do not violate the non-crossing condition. Only EPM $\mathcal{E}_3$ crosses $\mathcal{E}_1$. Note that an arc denotes a base pair within an EPM.

Formally, LCS-EPM is defined as follows. Given two nested RNAs $\mathcal{R}_1$, $\mathcal{R}_2$ and a set of exact pattern matchings $\mathbf{E}_{\gamma}^{1,2}$of these two RNAs, find an ordered matching $\mathcal{M}_{\mathsf{EPM}}$ consisting of a subset of EPMs from $\mathbf{E}_{\gamma}^{1,2}$ that has maximal cardinality. Thus, $\mathcal{M}_{\mathsf{EPM}}$ is defined as the union $\mathcal{M}_{\mathsf{EPM}}=\bigcup\mathcal{C}$ of a subset $\mathcal{C}\subseteq\mathbf{E}_{\gamma}^{1,2}$, where all EPMs contained in $\mathcal{C}$ are mutually non-crossing. Note that this implies that the found subsequence is a common subsequence since $\mathcal{M}_{\mathsf{EPM}}$ is an ordered matching. The common base pairs are induced by the EPMs itself.

Given a library of EPMs, our algorithm works by singling out the best combination of compatible EPMs. This task is performed efficiently by dynamic programming. The main idea is to recursively reduce the problem of solving the EPM puzzle for the EPMs enclosed in subsequences $S^1[i\ldots j]$ and $S^2[k\ldots l]$ to the problem for smaller subsequences. For our recursion scheme, we exploit the special structure of EPMs, which span matchings of certain subsequences of consecutive nucleotides. Between the boundaries of these matched consecutive subsequence, EPMs can omit subsequences; thereby they contain holes.

Figure 3 illustrates this structure of EPMs and shows how, given a single EPM $\mathcal{E}$, the relative position of the other EPMs to $\mathcal{E}$ can be distinguished. Formally, this is defined via the boundaries and holes of a single EPM.

2.3 Algorithmic concepts: boundaries and holes

The nucleotide positions of a pattern $\mathcal{P}$ of size k can be written as an increasing sequence. Similarly, an EPM $\mathcal{E}$ of size k over two RNAs is given with its corresponding patterns $\mathcal{P}_1$ in $\mathcal{R}_1$ and $\mathcal{P}_2$ in $\mathcal{R}_2$ and their increasing sequences $\mathcal{P}_1=\langle p_1,p_2,\ldots,p_k\rangle$ and $\mathcal{P}_2=\langle q_1,q_2,\ldots,q_k\rangle$.

2.3.1 Boundaries of EPMs In the view of the secondary structure, the elements (p_1,p_k) and (q_1,q_k) determine the outside borders of the EPM. Therefore, we call them *outside-boundaries* and write them as $\mathsf{OUT}_{\mathcal{E}}=\langle(p_1,p_k),(q_1,q_k)\rangle$. In the view of an arc-annotated sequence, we call (p_1,q_1) *left-outside-boundaries* and (p_k,q_k) *right-outside-boundaries* and denote them as $\mathsf{LEFT}_{\mathcal{E}}$ and $\mathsf{RIGHT}_{\mathcal{E}}$.

If an EPM contains base pairs, the structural shape is more complex and the outside-boundaries are not sufficient to describe all structural borders. If not all enclosed nucleotides of a base pair are part of the EPM, then there exist two positions in each RNA that form an additional structural border *inside* the range of the outside-boundaries. In addition, if a pattern contains several independent base pairs (e.g. in a multi-loop), there can be several such inside borders (cf. Fig. 4). The set of all such borders is called *inside-boundaries* and is defined as $\mathsf{IN}_{\mathcal{E}}=\{\langle(p_i,p_{i+1}),(q_j,q_{j+1})\rangle \mid p_{i+1}>p_i+1 \Leftrightarrow q_{j+1}>q_j+1\}$. Note that *outside-boundaries* always exists, whereas the set *inside-boundaries* can be empty. For example, assume an EPM that comprises only unpaired nucleotides or a complete hairpin including the closing bond. If an EPM consists of only one base pair in each sequence, then inside-and outside-boundaries are identical. With the superscript index for the RNA we retrieve the boundaries for a single RNA. For example $\mathsf{LEFT}^1_{\mathcal{E}}=p_1$.

2.3.2 Holes Holes are directly related to inside-boundaries and describe the subsequences which are not the part of the subsequence

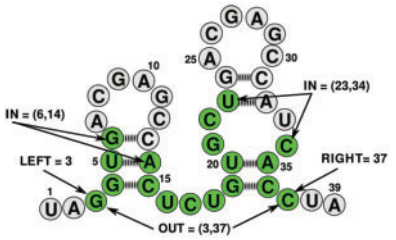

Fig. 4. A pattern of an EPM in one RNA (green nucleotides). The different boundaries are indicated.

$S_i[\mathsf{LEFT}^i_{\mathcal{E}}, \mathsf{RIGHT}^i_{\mathcal{E}}]$ of an EPM $\mathcal{E}$. For a given EPM $\mathcal{E}$ with its set of inside-boundaries $\mathsf{IN}_{\mathcal{E}}$, the set of holes with minimal size γ is defined as $\mathsf{HOLES}_{\mathcal{E}} = \{\langle (l^1, r^1), (l^2, r^2)\rangle \mid \ r^1 \geq l^1 + \gamma \ \wedge \ r^2 \geq l^2 + \gamma \}$. We introduce the notations h^{L1}, h^{R1}, h^{L1} and h^{R2} to refer to l^1, r^1, l^2 and r^2 of a hole $h = \langle (l^1, r^1), (l^2, r^2)\rangle$, respectively. For each $h \in \mathsf{HOLES}_{\mathcal{E}}$ there exists a pair of inside-boundaries with $\langle (h^{L1}-1, h^{R1}+1), (h^{L2}-1, h^{R2}+1)\rangle \in \mathsf{IN}_{\mathcal{E}}$. Clearly, a hole spans a substring $S_1[h^{L1}\ldots h^{R1}]$ in the first RNA and a substring $S_2[h^{L2}\ldots h^{R2}]$ in the second RNA. With γ we refer to the same size as indicated by $\mathbf{E}^{1,2}_{\gamma}$.

According to the length of the induced subsequences $S_i[h^{Li}\ldots h^{Ri}]$, we can sort all holes in one RNA. Let $h_i \in \mathsf{HOLES}_{\mathcal{E}_i}$ and $h_j \in \mathsf{HOLES}_{\mathcal{E}_j}$ two holes for any two $\mathcal{E}_i, \mathcal{E}_j \in \mathbf{E}^{1,2}_{\gamma}$. We define an ordering $h_i \preceq_{\mathsf{HOLES}} h_j$ in $\mathcal{R}_1$ if and only if h_i is of smaller size than h_j or of equal size in $\mathcal{R}_1$, i.e. $h_i \preceq_{\mathsf{HOLES}} h_j \Longleftrightarrow (h_i^{R1} - h_i^{L1}) \leq (h_j^{R1} - h_j^{L1})$.

2.4 Dynamic programming recursion for LCS-EPM

The essential difference of LCS-EPM to other alignment-based RNA comparison problems (including LAPCS) is that it treats a common substructure (i.e. an exact pattern matching) as a whole, unbreakable unit. This means that a solution of LCS-EPM either completely includes or completely excludes the edges (p,q) of each EPM. Following this idea, we want to compute the longest collinear sequence of EPMs which does not contain any crossing and overlapping EPMs.

The overall solution for LCS-EPM is constructed by a bottom-up approach from the comparison of substructures that are covered by the subsequences $S_1[i\ldots j]$ and $S_2[k\ldots l]$. In principle, this requires a four-dimensional matrix, denoted as $D(i,j,k,l)$, which contains the maximal score for combining EPMs that match only bases in $S_1[i\ldots j]$ and $S_2[k\ldots l]$. However, we can restrict ourselves to two-dimensional matrices using our notions of boundaries and holes for an exact pattern matching $\mathcal{E}$. For each hole, we introduce one two-dimensional matrix of entries $\mathbf{D}^h(j,l)$, such that $\mathbf{D}^h(j,l)$ is $D(h^{L1}, j, h^{L2}, l)$ of our imaginary four-dimensional matrix.

Finding non-crossing regions relative to an EPM is achieved as follows: all nucleotides before $\mathsf{LEFT}_{\mathcal{E}}$, i.e $S_i[1, \mathsf{LEFT}^i_{\mathcal{E}} - 1]$, as well as all nucleotides after the $\mathsf{RIGHT}_{\mathcal{E}}$, i.e. $S_i[\mathsf{RIGHT}^i_{\mathcal{E}} + 1, |S_i|]$ fulfill the non-crossing condition. This means that any EPM with its outside-boundaries $\mathsf{OUT}_{\mathcal{E}}$ in these regions is non-crossing relative to the considered EPM. Similarly we handle EPMs that contain base pairs with the introduced notion of $\mathsf{HOLES}_{\mathcal{E}}$. All EPMs that are located inside any hole of $\mathcal{E}$ cannot cross or overlap with $\mathcal{E}$.

The recursion scheme for a dynamic programming algorithm is as follows. Any $\mathcal{E}$ is handled only once at its right-outside-boundary $\mathsf{RIGHT}_{\mathcal{E}}$. The score of $\mathcal{E}$ is composed of the score *before* $\mathcal{E}$ (Fig. 3), given at the position $\mathsf{LEFT}_{\mathcal{E}} - 1$, plus the size of $\mathcal{E}$ itself, denoted by the function ω, plus possible scores between inside-boundaries, given recursively by the computation for scores for holes $h \in \mathsf{HOLES}_{\mathcal{E}}$. This last recursion case recurses to possible substructures and therefore suggests the use of a four-dimensional matrix. However, it suffices to use only quadratic space, since (1) all the scores for EPMs are stored in a vector with entries $\mathbf{S}_{\mathcal{E}}$ and (2) the score of each hole of an EPM can be computed using only a two-dimensional matrix. By ordering all holes according to $\preceq_{\mathsf{HOLES}}$, we guarantee that all necessary scores are already computed and stored, whenever an EPM is considered. Due to this order, the recursion starts with the smallest holes and goes on to the larger ones. Note that the two holes of the same size can be treated in any order.

For the formal description of the recursion, fix a hole h. The following recursion scheme works for any $h^{L1} \leq j \leq h^{R1}$ and $h^{L2} \leq l \leq h^{R1}$.

$$\mathbf{D}^h(j,l) = \max \begin{cases} \mathbf{D}^h(j-1,l) \\ \mathbf{D}^h(j,l-1) \\ \mathbf{D}^h(i-1,k-1) + \mathbf{S}_{\mathcal{E}}, \\ \quad \text{if } \exists \mathcal{E} \in \mathbf{E}^{1,2}_{\gamma} \text{ with } \mathsf{RIGHT}_{\mathcal{E}} = (j,l) \text{ and} \\ \qquad \mathsf{LEFT}_{\mathcal{E}} = (i,k), i \geq h^{L1}, k \geq h^{L2} \end{cases}$$

$$\mathbf{S}_{\mathcal{E}} = \omega(\mathcal{E}) + \sum_{h \in \mathsf{HOLES}_{\mathcal{E}}} \mathbf{D}^h(h^{R1}, h^{R2}).$$

After filling the matrices, the best score is computed from treating the whole sequence as hole. With a standard traceback technique the set of EPMs that form the LCS-EPM are found.

2.5 Complexity

Let $n = |S_1|$ and $m = |S_2|$ denote the lengths of the sequences. The time complexity depends primarily on the total number of holes. The set $\mathbf{E}^{1,2}_{\gamma}$ contains maximal $n \cdot m$ different holes which is estimated with $O(nm)$. The proof is omitted. For each hole, we fill a two-dimensional matrix with a size of at most $|S_1[l^1, r^1]| \leq |S_1| = n$ and $|S_2[l^2, r^2]| \leq |S_2| = m$. Consequently, for all holes we need $O(n^2m^2)$ time as worst case complexity. For real RNAs, a more appropriate time complexity can be given as $O(H \cdot nm)$ with H as the number of holes, since $H \ll n \cdot m$. This explains the fast running time of our algorithm on RNA. The space complexity is only $O(nm)$ because for each hole, after computing its score contribution and adding the score to its EPM, the space for the corresponding matrix $\mathbf{D}^h$ is recycled.

We summarize the complexity of solving the LCS-EPM problem as follows. Given two nested RNAs $\mathcal{R}_1 = (S_1, B_1)$ and $\mathcal{R}_2 = (S_2, B_2)$. The problem to determine the longest common subsequence of exact pattern matchings (LCS-EPM), including computation of $\mathbf{E}^{1,2}_{\gamma}$, is solvable in total $O(n^2m^2)$ time and $O(nm)$ space.

2.6 Speeding up RNA alignment by EPMs

One important application of LCS-EPM is the use of the predicted alignment edges $\mathcal{M}_{\mathsf{EPM}}$ as anchor constraints for sequence–structure alignment methods (Bauer *et al.*, 2007; Havgaard *et al.*, 2007; Will *et al.*, 2007). The idea of this combined alignment approach is to first solve the LCS-EPM for two given RNAs and then hand over the obtained result to an (usually much more expensive) sequence–structure alignment algorithm. This algorithm is used to fill the unaligned space between the exact pattern matchings in $\mathcal{M}_{\mathsf{EPM}}$ in order to produce a complete alignment, i.e. an alignment that also includes all the bases that do not occur in exact pattern matchings.

In general, anchor constraints restrict the space of possible alignments. Thus any alignment algorithm can be sped up by the use of such constraints. Therefore, one expects a speed up of the existing sequence–structure alignment tools that support anchor constraints, when one combines them with the preprocessing by `ExpaRNA` that generates anchor constraints. Thus, the proposed combination will result in an accelerated RNA alignment approach compared to the underlying RNA alignment approach alone, which will work for any available alignment method.

In particular, we modified the `LocARNA` algorithm for simultaneous folding and alignment of two RNA sequences S_1 and S_2 in order to profit from anchors. As a Sankoff-style algorithm, `LocARNA` essentially evaluates

Table 1. Comparison of the number of exactly matching alignment edges found by LCS-EPM and two alignment methods

Methods	IRES RNAs			16S rRNAs		
	No. of matches	Coverage (%)	Time (s)	No. of matches	Coverage (%)	Time
ExpaRNA	175	45	0.97	875	57	16.9 s
RNA_align	192	50	62.1	861	56	1 h 35 m
RNAforester	128	33	5.41	847	55	7 m 25 s
Comparison		IRES RNAs No. of common matches		16S rRNAs No. of common matches		
ExpaRNA and RNA_align		159 (82.8%)		688 (79.9%)		
ExpaRNA and RNAforester		103 (80.5%)		700 (82.6%)		

In the lower part, no. of *common matches* defines the number of identical aligned nucleotides of ExpaRNA and the other methods.

the recursion

$$M_{ij;kl} = \max \begin{cases} M_{ij-1;kl-1} + \sigma(j,l) \\ M_{ij-1;kl} + \alpha \\ M_{ij;kl-1} + \alpha \\ \max\limits_{j'l'} M_{ij'-1;kl'-1} + D_{j'j;l'l} \end{cases}$$

$$D_{ij;kl} = M_{ij-1;kl-1} + \tau_{ij;kl},$$

where i,j,k,l are sequence positions, i.e. $1 \le i < j \le n = |S_1|$ and $1 \le k < l \le m = |S_2|$, α is the gap cost, σ is a base similarity function and τ is a base pair similarity function τ, which reflects Turner's RNA energy model (Hofacker *et al.*, 2004; Mathews *et al.*, 1999). An entry $M_{ij;kl}$ contains the maximal score of alignments of $S_1[i..j]$ with $S_2[k..l]$, whereas for the entries $D_{ij;kl}$ the alignments additionally have to match the base pairs (i,j) and (k,l). In consequence, $D_{ij;kl}$ are only required when (i,k) and (j,l) can be alignment edges of some alignment at all. For computing all entries $D_{ij;kl}$ with a common (i,k), the algorithm fills the matrix slice $M_{i\cdot;k\cdot}$, which is the main load of the algorithm.

Given anchors, the algorithm can be modified to require less entries in $D_{ij;kl}$, namely only those where (i,k) and (j,l) are compatible with the anchors. Particularly, this implies that it needs to compute only entries $M_{ij;kl}$ where (i,k) is compatible with the anchor constraints.

For example, assume that we have a single anchor constraint $(n/2, m/2)$ (w.l.o.g. n and m even). Because only alignment edges (i,k) with $i \le n/2$ and $k \le n/2$ or $i > n/2$ and $k > n/2$ are compatible with the anchor, the algorithm computes only entries in $M_{ij;kl}$ for those (i,j), i.e. only half of the entries compared to the unconstrained algorithm.

3 RESULTS

We implemented the algorithm for finding the longest common subsequence of exact RNA patterns (i.e. LCS-EPM) in the tool ExpaRNA. The algorithm to determine all EPMs is implemented according to Backofen and Siebert (2007). ExpaRNA is implemented in C++.

We see at least two main application areas for ExpaRNA. First, given two RNAs along with their known or predicted secondary structure, the result of ExpaRNA comprises the optimal set of compatible exact common substructures. In biology, this can be used to get a good, first overview of existing similarities. Second, due to the fast running time of ExpaRNA, it is very attractive to use ExpaRNA for high-throughput RNA analysis tasks. We designed scenarios for both applications to study the different uses of our tool in detail.

3.1 Comparative structural analysis of large RNAs

Here, we study the application of ExpaRNA for analyzing large RNAs that are very costly to compare by other sufficiently accurate tools and where ExpaRNA elucidates information about identical structural motifs, which is not directly addressed by these tools and therefore may remain hidden. To enable an evaluation of our results, the experiments are performed on medium-sized and large RNAs where sequence–structure alignment tools are still applicable.

We have chosen two pairs of RNAs: (**a**) two IRES RNAs from hepatitis C virus, which belong both to the Rfam family HCV_IRES for IRESs (Griffiths-Jones *et al.*, 2005). GenBank: AF165050 (bases 1–379) and D45172 (bases 1–391). The secondary structures were predicted by RNAfold (Hofacker *et al.*, 1994). (**b**) Two 16S rRNAs. The first RNA is from *Escherichia coli* and is 1541 bases long. The second RNA of length 1551 stems from *Dictyostelium discoideum* (GenBank codes: J01859 and D16466). The secondary structures were taken from the Comparative RNA Web (CRW) site (Cannone *et al.*, 2002).

Table 1 shows the results for both pairs of RNAs. The solution of LCS-EPM is depicted as annotation of the secondary structures in Figure 5 for the IRES RNAs and in Figure 6 for the 16S rRNAs. These figures are directly produced by ExpaRNA using the Vienna RNA Package (Hofacker *et al.*, 1994) for the structure layout. For the IRES RNAs, the numbers mark the five largest EPMs from the set $\mathbf{E}_\gamma^{1,2}$ and correspond to the manually marked EPMs in Backofen and Siebert (2007). LCS-EPM predicts all of them automatically. In the case of the 16S rRNAs, the result of ExpaRNA shows significant similarities in nearly all stem and loop regions. Note that the set $\mathbf{E}_\gamma^{1,2}$ was computed with $\gamma = 2$ for both examples.

We compare our results with the output of RNA_align and RNAforester. The first method computes sequence-structure alignments according to the general edit distance algorithm (Jiang *et al.*, 2002). The RNAforester program of Höchsmann *et al.* (2003) is built upon the tree editing algorithm for ordered trees of Jiang *et al.* (1995) and extends it to calculate forest alignments. We compare us with these tools since both tools cover the state-of-the-art in RNA alignment that is based on fixed structures. The general edit

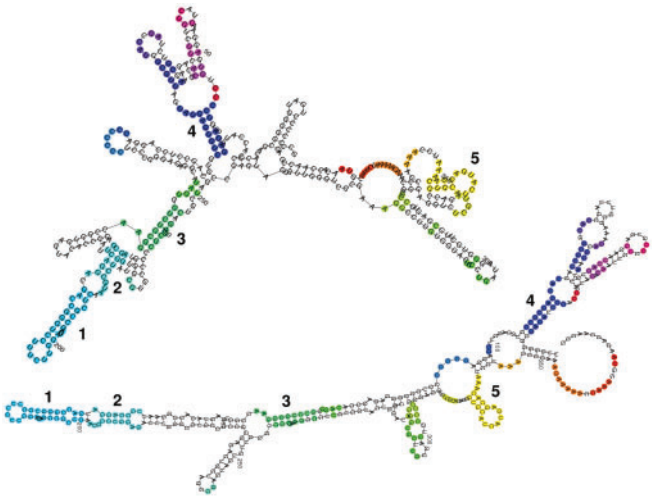

Fig. 5. LCS-EPM approach applied to two hepatitis C virus IRES RNAs. The colored nucleotides represent the found LCS-EPM with a coverage of 45% (175 nt). Each EPM is shown in a different color. The numbers indicate the five largest EPMs from $\mathbf{E}_{\gamma}^{1,2}$. GenBank: D45172 (upper RNA), AF165050 (lower RNA).

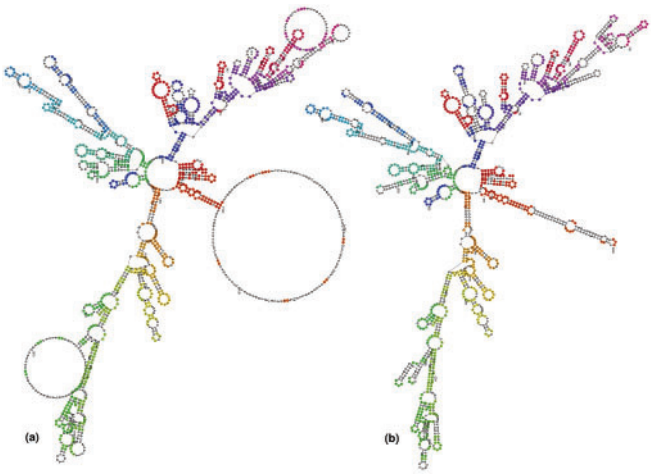

Fig. 6. LCS-EPM approach applied to two 16S RNAs. The colored nucleotides represent the found LCS-EPM with a coverage of 57% (875 nt). Each EPM is shown in a different color. (**a**) *D. discoideum* 16S rRNA (D16466), (**b**) *E. coli* 16S rRNA (J01859).

distance algorithm is a classic editing type algorithm for RNA comparison, whereas `RNAforester` represents the class of tree alignment-based algorithms, which can be due to their working principle much faster, but are less accurate than editing algorithms.

We compared the methods by the number of common realized alignment edges. Therefore, we have first computed the alignments for both RNA pairs. Next, we have counted all positions with exact sequence–structure matchings in these alignments and also determined the intersections with LCS-EPM. Note that the time for `ExpaRNA` in Table 1 includes the time to determine all EPMs for the two IRES RNAs (0.44s) and for the two 16S rRNAs (1.2s). The

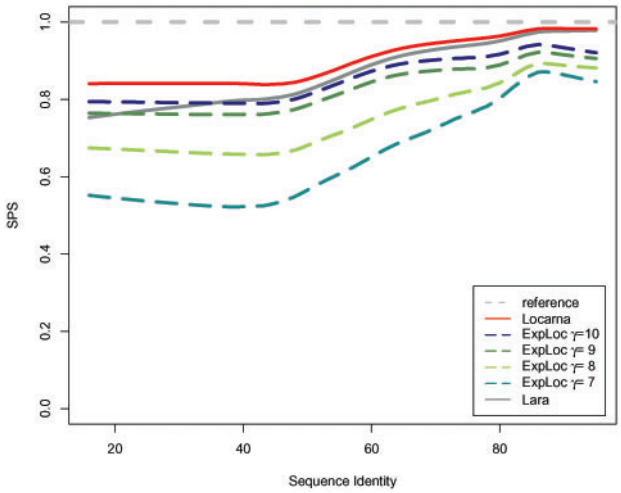

Fig. 7. Obtained alignment qualities for different minimal EPM sizes γ in comparison to `LocARNA` and `Lara` on `Bralibase 2.1` k2 dataset.

given sequence coverage rate is twice the number of predicted exact matches divided by the sum of the two sequence lengths.

3.2 Speeding up RNA alignment for large-scale analysis

Here, we study the performance of `ExpaRNA` for high-throughput RNA analysis. In Section 2.6, we showed by which means sequence–structure alignment algorithms can profit from anchor constraints and suggested to combine such tools with `ExpaRNA` that yields EPMs as anchor constraints in the form of a pre-computation step.

In order to assess the possible speedup by this combination, we tested `ExpaRNA` in combination with the `LocARNA` algorithm (Otto *et al.*, 2008; Will *et al.*, 2007).

The accuracy of our combined approach (called `ExpLoc`) was evaluated with the Bralibase 2.1 benchmark (Gardner *et al.*, 2005; Wilm *et al.*, 2006). The Bralibase 2.1 consists of a collection of hand-curated sets of RNA alignments. Because we are interested in the performance of pairwise alignment, we choose the k2 dataset with 8976 pairwise alignments. For each reference alignment, we compute the corresponding `ExpLoc` alignment and determined its sum of pair scores (SPS)/Compalign score (Bahr *et al.*, 2001; Gardner *et al.*, 2005; Wilm *et al.*, 2006) that measures the accuracy of reproducing the reference alignment. Furthermore, we recorded the running times of `ExpLoc` and `LocARNA` for each k2 alignment.

For the computation of a single `ExpLoc` alignment, we first computed the mfe structure with `RNAfold` of each sequence and input the two RNAs to `ExpaRNA`. Afterwards, the `ExpaRNA` output is used as anchor constraints for `LocARNA` in order to obtain the complete alignment of the two RNAs.

To test the performance of the two approaches, we carried out five experiments. First, we examined the accuracy of `LocARNA` alone. The other four experiments evaluate the performance of the combined approach `ExpLoc`. Here, we assessed the resulting alignment quality for different values $\gamma = 7, 8, 9$ and 10 for the `ExpaRNA` algorithm.

Figure 7 shows the achieved SPS scores at different levels of sequence identity for all five experiments. In addition, we

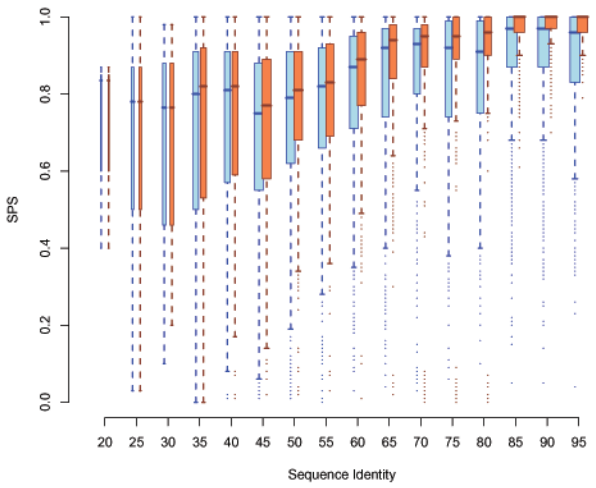

Fig. 8. Comparison of the quality of obtained results for ExpLoc (light blue) and LocARNA (orange). The boxplot shows distributions of sum-of-pairs scores (SPS) on the *y*-axis for different sequence identities on the *x*-axis for all 8976 pairwise alignments from Bralibase 2.1 and an minimal EPM size of $\gamma = 10$. To compute distributions, alignments were grouped according to their APSI in intervals of width 5.

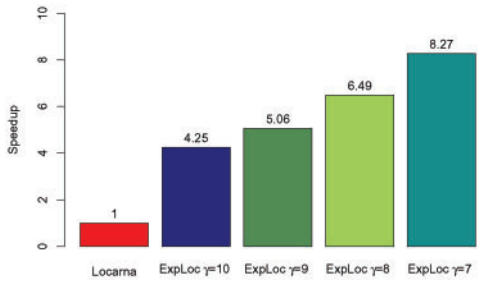

Fig. 9. Achieved speedup of ExpLoc with respect to LocARNA running time when using different minimal EPMsizes γ. Total times were measured for both methods when applied to all alignments of the Bralibase 2.1 k2 dataset.

included the performance of the Lara sequence–structure alignment algorithm (Bauer *et al.*, 2007).

Figure 8 shows a boxplot (also called box-and-whisker plot) visualizing min-values, max-values, medians and quartiles of the SPS/Compalign score distribution for varying pairwise sequence identities.

The obtained speedup factors shown in Figure 9 are calculated relative to the LocARNA algorithm. The shown values correspond to the experiments in Figure 7. The overall running time of LocARNA was 19 h 26 min. All computations were carried out on a Pentium 4 with 3.2 GHz.

4 DISCUSSION

Our results indicate that ExpaRNA can be advantageous in different application scenarios. In comparative RNA analysis, the results of ExpaRNA exhibit the existing similarities between RNA structures in a nice way. Existing relationships can be detected in a fraction of runtime without using a full alignment procedure.

Due to the availability of more and more large-scale datasets from modern pyro-sequencing techniques, high-throughput analysis methods for thousands of RNAs are needed. We analyzed the contribution of ExpaRNA for such tasks with the Bralibase benchmark. In general, our combined approach yields comparable results like other sequence–structure alignment algorithms. We observed a scaleable tradeoff between speedup and resulting alignment quality according to the selected minimal EPM size γ (Figs 7 and 9). By using different γ parameters our combined approach ExpLoc can be nicely balanced. This is important for problems with large datasets in which often a lower quality setting is sufficent. Moreover, our results show that anchor constraints are able to speedup Sankoff-style alignment algorithms in general (see Section 2.6).

A more fine-grained picture of the achieved accuracy of ExpLoc with $\gamma = 10$ is shown in Figure 8. In the $< 70\%$ sequence identity the differences are small. The lowered quality especially in region with a high sequence identity can be explained by the used mfe structures for ExpaRNA. Only slight differences in the sequence result in wide changes of the secondary structure which in turn leads to wrong predicted anchors. However, pure sequence alignment programs are sufficent here. For low sequence identities ($\leq 30\%$), there are nearly no differences. Here, ExpaRNA often does not find anchors which result in a standard LocARNA alignment. However, these cases are rare, which is also indicated by the width of the boxes in Figure 8.

The different speedups of ExpLoc for different γ values can be explained by the number of predicted anchor points. For $\gamma = 7$ there exists more anchors than for $\gamma = 10$. Further, we observe from our data speedups for short as well as for long alignments (Supplementary Figs 1 and 2). In particular, the speedup for long alignments is higher than for small ones, but also the majority of small alignments are accelerated. For longer RNAs we observe speedups around 100. We also look into the distribution of the speedups over different sequence identity classes. In general, sequences with a high sequence identity gain a higher speedup, but we also observe high speedups for classes between 35% and 65% sequence identity. This range is especially relevant for sequence–structure alignment methods, as pure sequence alignment methods will fail here.

Finally, we also observe 336 alignments for ExpLocwith $\gamma = 10$ (447 for $\gamma = 7$) resulting in a better SPS score than LocARNA alone.

5 CONCLUSION

We have developed a new algorithm for the pairwise sequence–structure comparison of RNAs and implemented it in the program ExpaRNA. Our approach utilizes common substructures for the detection of global similarities between two RNAs. We have applied the presented dynamic programming algorithm to two different kinds of application. In comparative sequence analysis, ExpaRNA can be used as good overview of existing similarities between two RNAs. Especially for large RNAs, ExpaRNA produces fast meaningful results without the need for usually more expensive alignment methods. In addition, we tested the performance of ExpaRNA in large-scale data analysis. Here, the main idea is to use the predicted LCS-EPM, i.e. an optimal set of compatible substructures, as anchor constraints for Sankoff-style alignment

algorithms in order to compute a complete gapped global alignment. We tested ExpaRNA in combination with the LocARNA algorithm on the Bralibase benchmark. In our experiments, we observe a trade-off between quality and speedup according to the chosen parameter γ. However, we get a speedup of 4.25 even in the highest tested quality setting, where the quality of the produced alignment is comparable to other sequence–structure alignment methods. The achieved results also suggests further exploration of the full potential of the ExpaRNA and ExpLoc approach for a variety of RNA structure comparsion-based applications.

Funding: German Research Foundation (DFG grant BA 2168/2-1 SPP 1258); Federal Ministry of Education and Research (BMBF grant 0313921 FORSYS/FRISYS).

Conflict of Interest: none declared.

REFERENCES

Allali,J. and Sagot,M.-F. (2005) A new distance for high level RNA secondary structure comparison. *IEEE/ACM Trans. Comput. Biol. Bioinfor.*, **2**, 3–14.

Altschul,S.F. *et al*. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.

Backofen,R. and Siebert,S. (2007) Fast detection of common sequence structure patterns in RNAs. *J. Discrete Algorithm*, **5**, 212–228.

Bafna,V. *et al*. (1995) Computing similarity between RNA strings. In *Proceedings of the 6th Symposium Combinatorial Pattern Matching*. Zvi,G. and Esko,U eds. *Lecture Notes in Computer Science*, pp. 1–16.

Bahr,A. *et al*. (2001) BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic Acids Res.*, **29**, 323–326.

Bauer,M. *et al*. (2007) Accurate multiple sequence-structure alignment of RNA sequences using combinatorial optimization. *BMC Bioinformatics*, **8**, 271.

Blin,G. *et al*. (2003) RNA sequences and the EDIT(NESTED,NESTED) problem. Technical Report RR-IRIN-03.07, IRIN, Université de Nantes.

Cannone,J.J. *et al*. (2002) The Comparative RNA Web (CRW) Site: an online database of comparative sequence and structure information for ribosomal, intron, and other RNAs: Correction. *BMC Bioinformatics*, **3**, 15.

Evans,P.A. (1999) *Algorithms and Complexity for Annotated Sequence Analysis*. Ph.D. thesis, University of Alberta.

Gardner,P.P. *et al*. (2005) A benchmark of multiple sequence alignment programs upon structural RNAs. *Nucleic Acids Res.*, **33**, 2433–2439.

Griffiths-Jones,S. *et al*. (2005) Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res.*, **33**, D121–D124.

Havgaard,J.H. *et al*. (2007) Fast pairwise structural RNA alignments by pruning of the dynamical programming matrix. *PLoS Comput. Biol.*, **3**, 1896–1908.

Hentze,M.W. and Kuhn,L.C. (1996) Molecular control of vertebrate iron metabolism: mRNA-based regulatory circuits operated by iron, nitric oxide, and oxidative stress. *Proc. Natl. Acad. Sci. USA*, **93**, 8175–8182.

Höchsmann,M. *et al*. (2003) Local similarity in RNA secondary structures. In *Proceedings of Computational Systems Bioinformatics (CSB 2003)*, vol. 2. IEEE Computer Society, pp. 159–168.

Hofacker,I.L. *et al*. (1994) Fast folding and comparison of RNA secondary structures. *Monatsh. Chem.*, **125**, 167–188.

Hofacker,I.L. *et al*. (2004) Alignment of RNA base pairing probability matrices. *Bioinformatics*, **20**, 2222–2227.

Huttenhofer,A. *et al*. (1996) Solution structure of mRNA hairpins promoting selenocysteine incorporation in Escherichia coli and their base-specific interaction with special elongation factor SELB. *RNA*, **2**, 354–366.

Jiang,T. *et al*. (1995) Alignment of trees - an alternative to tree edit. *Theor. Comput. Sci.*, **143**, 137–148.

Jiang,T. *et al*. (2002) A general edit distance between RNA structures. *J. Comput. Biol.*, **9**, 371–388.

Lin,G. *et al*. (2002) The longest common subsequence problem for sequences with nested arc annotations. *J. Comput. Syst. Sci.*, **65**, 465–480.

Martineau,Y. *et al*. (2004) Internal ribosome entry site structural motifs conserved among mammalian fibroblast growth factor 1 alternatively spliced mRNAs. *Mol. Cell Biol.*, **24**, 7622–7635.

Mathews,D.H. and Turner,D.H. (2002) Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. *J. Mol. Biol.*, **317**, 191–203.

Mathews,D. *et al*. (1999) Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol.*, **288**, 911–940.

Otto,W. *et al*. (2008) Structure local multiple alignment of RNA. In *Proceedings of German Conference on Bioinformatics (GCB'2008)*, Vol. P-136 of *LNI*. Gesellschaft für Informatik, pp. 178–188.

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

Serganov,A. and Patel,D.J. (2007) Ribozymes, riboswitches and beyond: regulation of gene expression without proteins. *Nat. Rev. Genet.*, **8**, 776–790.

Torarinsson,E. *et al*. (2007) Multiple structural alignment and clustering of RNA sequences. *Bioinformatics*, **23**, 926–932.

Will,S. *et al*. (2007) Inferring non-coding RNA families and classes by means of genome-scale structure-based clustering. *PLOS Comput. Biol.*, **3**, e65.

Wilm,A. *et al*. (2006) An enhanced RNA alignment benchmark for sequence alignment programs. *Algorithms Mol. Biol.*, **1**, 19.

Wilting,R. *et al*. (1997) Selenoprotein synthesis in archaea: identification of an mRNA element of Methanococcus jannaschii probably directing selenocysteine insertion. *J. Mol. Biol.*, **266**, 637–641.

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