Fast RNA Structure Alignment for Crossing Input Structures

Rolf Backofen[1], Gad M. Landau[2], Mathias Möhl[3], Dekel Tsur[4], and Oren Weimann[5]

[1] Bioinformatics, Institute of Computer Science, Albert-Ludwigs-Universität, Freiburg, Germany. backofen@informatik.uni-freiburg.de
[2] Department of Computer Science, Haifa University, Haifa 31905, Israel, landau@cs.haifa.ac.il; Department of Computer and Information Science, Polytechnic Institute of NYU, Six MetroTech Center, Brooklyn, NY 11201-3840.
[3] Programming Systems Lab, Saarland University, Saarbrücken, Germany. mmohl@ps.uni-sb.de
[4] Ben-Gurion University, Beer-Sheva, Israel. dekelts@cs.bgu.ac.il
[5] Massachusetts Institute of Technology, Cambridge, MA 02139, USA. oweimann@mit.edu

Abstract. The complexity of pairwise RNA structure alignment depends on the structural restrictions assumed for both the input structures and the computed consensus structure. For arbitrarily crossing input and consensus structures, the problem is NP-hard. For non-crossing consensus structures, Jiang et al's algorithm [1] computes the alignment in $O(n^2m^2)$ time where n and m denote the lengths of the two input sequences. If also the input structures are non-crossing, the problem corresponds to tree editing which can be solved in $O(m^2n(1 + \log \frac{n}{m}))$ time [2]. We present a new algorithm that solves the problem for d-crossing structures in $O(dm^2n \log n)$ time, where d is a parameter that is one for non-crossing structures, bounded by n for crossing structures, and much smaller than n on most practical examples. Crossing input structures allow for applications where the input is not a fixed structure but is given as base-pair probability matrices.

Keywords: RNA, sequence structure alignment, simultaneous alignment and folding

1 Introduction

With the recent focus on non-protein-coding RNA (ncRNA) genes, interest in detecting novel ncRNAs has rapidly emerged. A recent screen on ncRNAs has detected more than 30000 putative ncRNAs in human genome [3], most of them with unknown function. Since the structure of RNA is evolutionarily more conserved than its sequence, predicting the RNA's secondary structure is the most important step towards its functional analysis [4].

The secondary structure of an RNA molecule can be calculated from its nucleotide sequence by determining a folding with minimal free energy [5–9]. Albeit this so-named thermodynamic approach is a success story in the analysis of RNA, it is known that predicting the secondary structure from a single sequence is error-prone, where the best available approaches can correctly predict only up to 73% of the base-pairs [10]. This situation can be improved by taking phylogenetic information into account, i.e., by predicting a common consensus structure from a whole set of evolutionary related RNA sequences.

There are several approaches for this problem (see [11] for an overview) which increase both in computational complexity as well as in the average quality. The simplest and fastest approach is to align the RNA sequences using a multiple sequence alignment, and then to fold the complete alignment using approaches like RNAalifold [12] and Petfold [13]. This has time complexity of $O(k^2n^2)$ for the pairwise alignment, and $O(n^3)$ for the final folding, which has to be applied only once on the complete alignment, where k is the number of sequences.

The second approach is to predict for all k sequences the minimum free-energy structure (with complexity $O(n^3)$), and then to perform progressive sequence-structure alignment whose complexity is dominated by the pairwise alignment steps. For a long time, the best complexity known for the pairwise alignment step was $O(n^3 \log(n))$ as given by the seminal work of Klein [14]. Just recently this has been improved to $O(n^3)$ [2]. However, this approach crucially depends on the quality of the initial structure prediction, which is error-prone.

Hence, the gold standard are Sankoff-like approaches [15–19] which *simultaneously* align and fold the sequences. However, as stated in [11], the Sankoff-approach requires "extreme amounts of memory and space" with a space complexity of $O(n^4)$ and a time complexity of $O(n^6)$. In [19], we improved this complexity to $O(n^4)$ time by aligning base-pair probability matrices. Basically, one is given two sets of weighted base-pairs that are possibly crossing, and the goal is to find the best common *nested* consensus structure for both sets, taking both base-pair weights and the associated RNA sequences into account.

In this work, we want to shorten the gap between sequence-structure alignment methods (with a complexity of $O(n^3)$), and the Sankoff-like approaches (with a complexity of $O(n^4)$ for alignment of base-pair probability matrices) for a practical application scenario. Basically, sequence-structure alignment approaches use exactly *one* structure per sequence as an input, whereas Sankoff-like approaches use *all* possible structures as an input. However, in many practical cases, one has a mixture of both, namely a main structure that allows for a small deviation. As shown in the example in Fig. 1, the alternative structures together form a crossing input structure, where the offset between crossing arcs is small. In this paper, we introduce a measurement for this deviation (d-crossing), and introduce an efficient algorithm with complexity $O(n^3 \log(n))$ given that the deviation is small (i.e., that the input base-pair probability matrix is d-crossing for a small constant d). Note that the crossing structure in Fig. 1 forms a two-page embedding (or is 2-colorable, as it is called in [20]), but our approach is not restricted to this class of structures.

The fast available sequence structure alignment methods for non crossing input structures as in Klein [14] (a formal definition of non-crossing is given in Section 2) rely on a heavy path decomposition which was so far only available for tree-like structures. Our approach generalizes this to d-crossing structures.

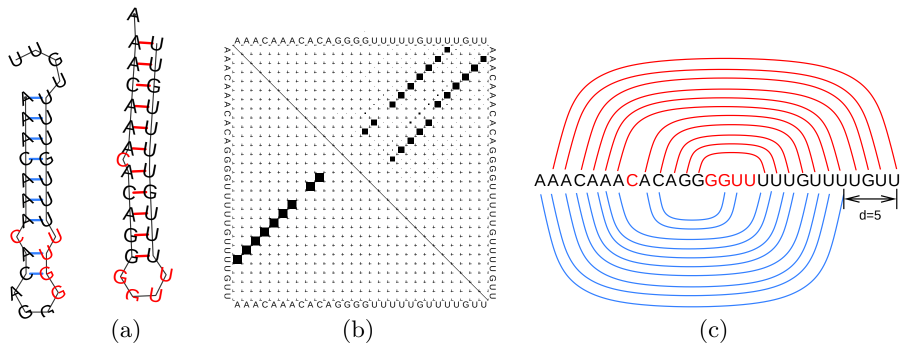

Fig. 1. (a) Two structures for the sequence AAACAAACACAGGGGUUUUUGUUUUGUU with similar free energy. The stem in the second sequence is shifted by 5 nucleotides. (b) associated base-pair probability matrix (upper triangle) and minimum free energy structure (lower triangle). The shifted stem is indicated by two parallel diagonals, a pattern often seen in RNA-structures. (c) both nested structures together form a crossing input. The outermost arcs of both structures are d-crossing for $d = 5$.

2 Preliminaries

An *arc-annotated sequence* is a pair (S, P), where S is a string over the set of bases $\{A, U, C, G\}$ and P is a set of arcs (l, r) with $1 \leq l < r \leq |S|$ representing bonds between bases. We allow more than one arc to be adjacent to one base, but require that $|P| \in O(|S|)$, that is, on average each base is adjacent to only a constant number of arcs. We denote the i-th symbol of S by $S[i]$ and the substring from symbol i to symbol j with $S[i \dots j]$. For an arc $p = (l, r)$, we denote its left end l and right end r by p^{L} and p^{R}, respectively. The *span* of p is defined as $\mathrm{span}(p) = p^{\mathrm{R}} - p^{\mathrm{L}} + 1$.

Two arcs p_1 and p_2 in an arc-annotated sequence (S, P) are *crossing* if $p_1^{\mathrm{L}} \le p_2^{\mathrm{L}} \le p_1^{\mathrm{R}} \le p_2^{\mathrm{R}}$ or $p_2^{\mathrm{L}} \le p_1^{\mathrm{L}} \le p_2^{\mathrm{R}} \le p_1^{\mathrm{R}}$. Two crossing arcs p_1 and p_2 are *d-crossing* if $|p_1^{\mathrm{L}} - p_2^{\mathrm{L}}| < d$ and $|p_1^{\mathrm{R}} - p_2^{\mathrm{R}}| < d$. An arc p_1 is *nested* in an arc p_2 if $p_2^{\mathrm{L}} < p_1^{\mathrm{L}} < p_1^{\mathrm{R}} < p_2^{\mathrm{R}}$. An arc p_1 *precedes* an arc p_2 if $p_1^{\mathrm{R}} < p_2^{\mathrm{L}}$. For every two arcs, either the two arcs are crossing, one of the arc is nested in the other, or one of the arc precedes the other. An arc-annotated sequence (S, P) containing crossing arcs is called *crossing*, otherwise *non-crossing* or *nested*. A *d-crossing sequence* is a crossing sequence in which every two crossing arcs are d-crossing.

3 Problem Definition

An *alignment* A of two arc-annotated sequences (S_1, P_1) and (S_2, P_2) is a set $A = A_{\text{match}} \uplus A_{\text{gap}}$. The set $A_{\text{match}} \subseteq [1, n] \times [1, m]$ of *match edges* satisfies that for all $(i, j), (i', j') \in A$, (1) $i > i'$ implies $j > j'$, and (2) $i = i'$ if and only if $j = j'$. Given A_{match}, the set of *gap edges* is implied as $A_{\text{gap}} := \{\, (i, -) \mid i \in [1, n] \wedge \nexists j.(i, j) \in A_{\text{match}} \,\} \cup \{\, (-, j) \mid j \in [1, m] \wedge \nexists i.(i, j) \in A_{\text{match}} \,\}$. A *consensus structure* for an alignment A is a matching $P \subseteq P_1 \times P_2$ that satisfies $(p_1, p_2) \in P \Rightarrow (p_1^{\mathrm{L}}, p_2^{\mathrm{L}}) \in A \wedge (p_1^{\mathrm{R}}, p_2^{\mathrm{R}}) \in A$. We require a consensus structure to be non-crossing (formally $\{(p_1, p_2), (p_1', p_2')\} \subseteq P \Rightarrow p_1$ and p_1' do not cross) and such that each base is adjacent to at most one arc (i.e. $\{(p_1, p_2), (p_1', p_2')\} \subseteq P \Rightarrow (p_1^{\mathrm{L}} = p_1'^{\mathrm{L}} \Leftrightarrow p_1^{\mathrm{R}} = p_1'^{\mathrm{R}})$).

Each alignment together with some consensus structure has an associated cost based on functions $\gamma_1 \in [1, n] \to \mathbb{N}$, $\gamma_2 \in [1, m] \to \mathbb{N}$, $\beta \in [1, n] \times [1, m] \to \mathbb{N}$, and $\alpha \in ([1, n])^2 \times ([1, m])^2 \to \mathbb{N}$. $\gamma_k(i)$ denotes the cost to align position i of sequence k to a gap, $\beta(i, j)$ the cost for a base match, i.e. cost to align position i of the first sequence to position j of the second sequence, provided arcs adjacent to i and j are not contained in the consensus structure, and $\alpha(p_a, p_b)$ denotes the cost to match arcs p_a, p_b in the consensus structure. The cost of an alignment A with consensus structure P, denoted $C_P(A)$, is

$$\sum_{(i,-) \in A} \gamma_1(i) + \sum_{(-,j) \in A} \gamma_2(j) + \sum_{((i,j),(i',j')) \in P} \alpha((i, j), (i', j')) + \sum_{(i,j) \in A'} \beta(i, j),$$

where A' is the set of all edges $(i, j) \in A$ such that there is no edge $(i', j') \in A$ for which $((i, j), (i', j')) \in P$ or $((i', j'), (i, j)) \in P$. Note that this scoring scheme can easily be instantiated with the edit distance scoring scheme of Jiang et al [1] if each base is adjacent to at most one arc. For this case we set $\gamma_1(i) = w_d + \psi_1(i)(\frac{w_r}{2} - w_d)$, $\gamma_2(j) = w_d + \psi_2(j)(\frac{w_r}{2} - w_d)$, $\beta(i,j) = \chi(i,j)w_m + (\psi_1(i) + \psi_2(j))\frac{w_b}{2}$, and $\alpha((i,j),(i',j')) = (\chi(i,j) + \chi(i',j'))\frac{w_{am}}{2}$ where ψ_1, ψ_2, χ, w_d, w_r, w_m, w_b, and w_{am} are defined as in [1]. However, we formulate the algorithm with the more general scoring scheme, since $\alpha((i,j),(i',j'))$ can be used to encode base pair weights which is more suitable in the presence of several adjacent arcs per base that represent alternative structures.

The *RNA structure alignment problem* is given two arc-annotated sequences (S_1, P_1) and (S_2, P_2), to find an alignment A and a consensus structure P such that $C_P(A)$ is minimal. For the remainder of this paper we fix two arc-annotated sequences (S_1, P_1) and (S_2, P_2) with $|S_1| = n$, $|S_2| = m$, $|P_1| \in O(n)$ and $|P_2| \in O(m)$ and assume that (S_1, P_1) is $d-$crossing. We assume w.l.o.g. that P_1 contains an arc $(1, n)$.

Arc annotated sequences are often classified as PLAIN, NEST, CROSS or UNLIM, as originally proposed in [21]. We solve for our scoring scheme the edit problem for a class that fully contains EDIT(NEST,NEST) and partially contains EDIT(UNLIM,UNLIM) (namely those instances where one structure is $d-$crossing and where on average each base is adjacent to only a constant number of arcs).

4 The Algorithm Recursion

The algorithm consists of two stages. The first stage computes the optimal costs to align certain fragments that are required for the second stage.

4.1 Stage 1

In the first stage, we compute a table M analogously to the recursion of Jiang et al. [1]. The entry $M[i, i', j, j']$ represents the minimal cost of an alignment between $(S_1[i \ldots i'], P_1 \cap [i, i']^2)$ and $(S_2[j \ldots j'], P_2 \cap [j, j']^2)$.

The base cases where $i' = i - 1$ and $j' = j - 1$ are initialized with $M[i, i-1, j, j-1] = 0$, the other entries are computed recursively as defined in Fig. 2. In the recursive computation, cases that rely on invalid items (i.e. where any of i, i', j, j' are not within their allowed range) are implicitly skipped. While Jiang et al's algorithm computes the entire alignment based on this recursion, we only compute entries of M for short fragments of the first sequence that have a length of at most $2d+2$, i.e. for $1 \leq i \leq n$, $i - 1 \leq i' \leq \min(i + 2d + 1, n)$, $1 \leq j \leq m$, and $j - 1 \leq j' \leq m$.

$$M[i,i',j,j'] = \min \begin{cases} M[i,i'-1,j,j'] + \gamma_1(i') & \text{I} \\ M[i,i',j,j'-1] + \gamma_2(j') & \text{II} \\ M[i,i'-1,j,j'-1] + \beta(i',j') & \text{III} \\ \text{for all } p_1 = (i_0,i') \in P_1,\ p_2 = (j_0,j') \in P_2 \text{ with } i \le i_0,\ j \le j_0 & \text{IV} \\ \quad M[i,i_0-1,j,j_0-1] + M[i_0+1,i'-1,j_0+1,j'-1] + \alpha(p_1,p_2) & \end{cases}$$

Fig. 2. Recursion for the table M.

4.2 Stage 2

For non-crossing input structures, the correspondence of these structures to trees allows for alignment methods that are asymptotically faster than the recursion used in the first stage [2, 14]. In our approach we apply a similar technique, but since our input structures do not correspond to trees, we select a subset $P_T \subseteq P_1$ of the arcs.

The arcs in P_T do not cross and at most one of them is adjacent to each base. Hence, the arcs in P_T form a tree structure that guides the recursive decomposition during the computation of the alignment.

Construction of $\boldsymbol{P_T}$ Define the *inner d-range* of p as $I_d(p) = [p^{\mathrm{L}}+1, p^{\mathrm{L}}+d-1] \times [p^{\mathrm{R}}-d+1, p^{\mathrm{R}}-1]$. For a set of arcs $P \subseteq P_1$, the set $\mathrm{tree}\,(P)$ is defined recursively as follows. If $P = \emptyset$ or all arcs in P have span at most $2d$ then $\mathrm{tree}\,(P) = \emptyset$. Otherwise, let p be some arc in P with maximum span (ties are broken arbitrarily), and

$$\begin{aligned} \mathrm{tree}\,(P) = & \{p\} \cup \mathrm{tree}\left(P \cap [1, p^{\mathrm{L}}-1]^2\right) \cup \mathrm{tree}\left(P \cap [p^{\mathrm{R}}+1, n]^2\right) \cup \\ & \mathrm{tree}\left((P \cap [p^{\mathrm{L}}+1, p^{\mathrm{R}}-1]^2) \setminus I_d(p)\right). \end{aligned}$$

Lemma 1. *Every arc in P crosses at most one arc in* $\mathrm{tree}\,(P)$.

Proof. Let p_1 and p_2 be two arcs in $\mathrm{tree}\,(P)$, and assume w.l.o.g. that $p_1^{\mathrm{L}} < p_2^{\mathrm{L}}$. We have that either p_2 is nested in p_1 or p_1 precedes p_2.

If p_2 is nested in p_1 then by the definition of $\mathrm{tree}\,(P)$, either $p_2^{\mathrm{L}} - p_1^{\mathrm{L}} \ge d$ or $p_1^{\mathrm{R}} - p_2^{\mathrm{R}} \ge d$. Suppose w.l.o.g. that $p_2^{\mathrm{L}} - p_1^{\mathrm{L}} \ge d$. Let p be an arc that crosses p_1. If $p^{\mathrm{L}} \le p_1^{\mathrm{L}}$ then $|p^{\mathrm{L}} - p_2^{\mathrm{L}}| \ge p_2^{\mathrm{L}} - p_1^{\mathrm{L}} \ge d$, so p does not cross p_2. If $p^{\mathrm{L}} > p_1^{\mathrm{L}}$ then $p^{\mathrm{L}} \le p_1^{\mathrm{L}} + d - 1 < p_2^{\mathrm{L}}$ and $p^{\mathrm{R}} \ge p_1^{\mathrm{R}} > p_2^{\mathrm{R}}$. Therefore, p_2 is nested in p, and in particular, p does not cross p_2.

If p_1 precedes p_2 then $p_2^{\mathrm{L}} > p_1^{\mathrm{R}} = p_1^{\mathrm{L}} + \mathrm{span}(p_1) - 1 \ge p_1^{\mathrm{L}} + 2d$. Therefore, for every arc p, either $|p^{\mathrm{L}} - p_1^{\mathrm{L}}| \ge d$, or $|p^{\mathrm{L}} - p_2^{\mathrm{L}}| \ge d$. We conclude that p cannot cross both p_1 and p_2. $\square$

Lemma 2. *An arc $p \in P$ satisfies $p \in I_d(p')$ for at most one arc $p' \in$* $\text{tree}\,(P)$*. If p does not cross an arc in* $\text{tree}\,(P)$ *then $p \in I_d(p')$ for a unique arc $p' \in$* $\text{tree}\,(P)$*.*

Proof. To prove the first part of the lemma, let p_1 and p_2 be two arcs in $\text{tree}\,(P)$ with $p_1^{\text{L}} < p_2^{\text{L}}$. Either p_2 is nested in p_1 or p_1 precedes p_2. If p_2 is nested in p_1 then either $p_2^{\text{L}} - p_1^{\text{L}} \geq d$ or $p_1^{\text{R}} - p_2^{\text{R}} \geq d$. In the former case, the intervals $[p_1^{\text{L}}+1, p_1^{\text{L}}+d-1]$ and $[p_2^{\text{L}}+1, p_2^{\text{L}}+d-1]$ are disjoints, and therefore $I_d(p_1) \cap I_d(p_2) = \phi$. Similarly, $I_d(p_1) \cap I_d(p_2) = \phi$ when $p_1^{\text{R}} - p_2^{\text{R}} \geq d$ or when p_1 precedes p_2. Thus, p cannot be both in $I_d(p_1)$ and $I_d(p_2)$.

We prove the second part of the lemma using induction on $|P|$. Let $P \subseteq P_1$ be a nonempty set of arcs, and let p be some arc in P that does not cross an arc in $\text{tree}\,(P)$. Let p' be the maximum span arc in P that is chosen when computing $\text{tree}\,(P)$. Recall that $\text{tree}\,(P) = \{p'\} \cup \text{tree}\,(P^1) \cup \text{tree}\,(P^2) \cup \text{tree}\,(P^3)$ where $P^1 = P \cap [1, p'^{\text{L}} - 1]^2$, $P^2 = P \cap [p'^{\text{R}} + 1, n]^2$, and $P^3 = (P \cap [p'^{\text{L}} + 1, p'^{\text{R}} - 1]^2) \setminus I_d(p')$. If $p \in I_d(p')$ we are done. Otherwise, since p does not cross p' and $p \notin I_d(p')$, we have that p is in some set P^i. Since $|P^i| < |P|$, by the induction hypothesis there is an arc $p'' \in \text{tree}\,(P^i)$ such that $p \in I_d(p'')$. □

We define $P_T = \text{tree}\,(P_1)$, and we call the arcs in P_T *tree arcs*. For every $p \in P_1$ we define $T(p)$ to be the unique tree arc p' such that p crosses p', if such arc exists. Otherwise, $T(p)$ is the unique tree arc p' such that $p \in I_d(p')$.

Lemma 3. *For every $p \in P_1$, $|p^L - T(p)^L| < d$ and $|p^R - T(p)^R| < d$.*

Proof. If p crosses $T(p)$ then the inequalities of the lemma are satisfied since (S_1, P_1) is d-crossing. Otherwise, from Lemma 2, $p \in I_d(T(p))$, and the inequalities of the lemma are satisfied by the definition of $I_d(\cdot)$. □

Lemma 4. *Let $p \in P_1$ and let $p' \in P_T$ such that $p' \neq p$ and p' is nested in $T(p)$. Then, p' is nested in p.*

Proof. Let p and p' be two arcs satisfying the conditions of the lemma. From the definition of $T(\cdot)$, p cannot cross p'. Moreover, from Lemma 3 and the fact that $\text{span}(p) > 2d$, p' cannot precede p, or vice versa. □

For every tree arc $p \in P_T$ we select a tree arc denoted $\text{hchild}(p)$ such that $\text{hchild}(p)$ is nested in p and $\text{span}(\text{hchild}(p))$ is maximum (if there is such an arc). For $p \in P_T$ and $p \neq (1, n)$, define $\text{parent}(p)$ to be the minimum span tree arc that p is nested in. We define $\text{parent}((1, n)) = (1, n)$.

Recursion For each $p \in P_T$ we build two tables L^p and R^p. Intuitively, one obtains the optimal alignments of the area below p or any arc crossing p by first extending the optimal alignments of $\mathrm{hchild}(p)$ or any arc crossing $\mathrm{hchild}(p)$ to the left (with L^p) and then to the right (with R^p). We compute the tables in an order such that for each p, L^p is computed before R^p and such that the tables of all $p' \in P_T$ that are nested in p are computed before the tables of p.

The table entries $L^p[i, i', j, j']$ and $R^p[i, i', j, j']$ have the same semantics as $M[i, i', j, j']$ and only differ in the domains of the indices i, i', j, j' and the recursions according to which they are computed. Let us first assume that $\mathrm{hchild}(p)$ is defined for p. Then, $L^p[i, i', j, j']$ is defined for

$$\begin{aligned}
\max(p^{\mathrm{L}} - d, \mathrm{parent}(p)^{\mathrm{L}}) \leq i \leq \mathrm{hchild}(p)^{\mathrm{L}}& \\
\mathrm{hchild}(p)^{\mathrm{R}} \leq i' \leq \min(\mathrm{hchild}(p)^{\mathrm{R}} + d, p^{\mathrm{R}})& \\
1 \leq j \leq m& \\
j - 1 \leq j' \leq m.&
\end{aligned}$$

and for $R^p[i, i', j, j']$ the domains of j and j' are the same, but i and i' must satisfy

$$\begin{aligned}
\max(p^{\mathrm{L}} - d, \mathrm{parent}(p)^{\mathrm{L}}) \leq i \leq \min(p^{\mathrm{L}} + d, \mathrm{hchild}(p)^{\mathrm{L}})& \\
\mathrm{hchild}(p)^{\mathrm{R}} \leq i' \leq \min(p^{\mathrm{R}} + d, \mathrm{parent}(p)^{\mathrm{R}}).&
\end{aligned}$$

If $\mathrm{hchild}(p)$ is not defined for p, no L^p table is computed and the R^p tables contain entries for

$$\begin{aligned}
\max(p^{\mathrm{L}} - d, \mathrm{parent}(p)^{\mathrm{L}}) \leq i \leq p^{\mathrm{L}} + d& \\
p^{\mathrm{L}} + d \leq i' \leq \min(p^{\mathrm{R}} + d, \mathrm{parent}(p)^{\mathrm{R}})&
\end{aligned}$$

and j, j' restricted as in the table R^p in the case where $\mathrm{hchild}(p)$ is defined. The domains of i and i' for the different cases are visualized in Fig. 3.

Computation of L^p All entries $L^p[i, i', j, j]$ with $i \geq \max(\mathrm{hchild}(p)^{\mathrm{L}} - d, p^{\mathrm{L}})$ are initialized as $L^p[i, i', j, j'] = R^{\mathrm{hchild}(p)}[i, i', j, j']$. All other entries are computed according to the recursion shown in Fig. 5. Again cases relying on invalid items are implicitly skipped. The last three cases of the recursion are visualized in Fig. 4.

Computation of R^p The computation of the R^p tables is similar to the computation of the L^p tables, only that the fragments are extended to

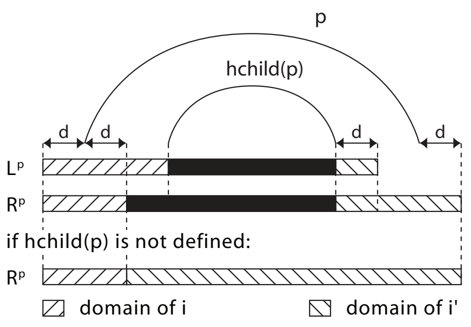

Fig. 3. Visualization of the domains for the different tables.

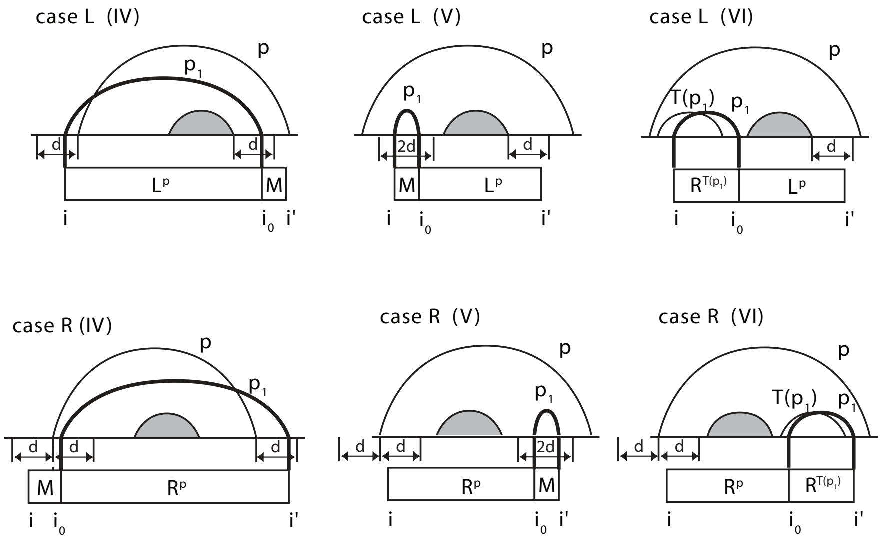

Fig. 4. Visualization of the recursion cases. The arc bounding the gray area denotes hchild(p).

$$L^p[i,i',j,j'] =$$
$$\min\begin{cases} L^p[i+1,i',j,j'] + \gamma_1(i) & \text{I} \\ L^p[i,i',j+1,j'] + \gamma_2(j) & \text{II} \\ L^p[i+1,i',j+1,j'] + \beta(i,j) & \text{III} \\ \text{for all } p_1=(i,i_0)\in P_1,\ p_2=(j,j_0)\in P_2 \text{ with } i_0\le i',\ j_0\le j', & \\ \text{and hchild}(p) \text{ is nested in } p_1 & \text{IV} \\ \quad L^p[i+1,i_0-1,j+1,j_0-1] + M[i_0+1,i',j_0+1,j'] + \alpha(p_1,p_2) & \\ \text{for all } p_1=(i,i_0)\in P_1,\ p_2=(j,j_0)\in P_2 \text{ with } i_0\le i',\ j_0\le j', & \\ \text{hchild}(p) \text{ is not nested in } p_1 \text{, and span}(p_1)\le 2d & \text{V} \\ \quad M[i+1,i_0-1,j+1,j_0-1] + L^p[i_0+1,i',j_0+1,j'] + \alpha(p_1,p_2) & \\ \text{for all } p_1=(i,i_0)\in P_1,\ p_2=(j,j_0)\in P_2 \text{ with } i_0\le i',\ j_0\le j', & \\ \text{hchild}(p) \text{ is not nested in } p_1 \text{, and span}(p_1)> 2d & \text{VI} \\ \quad R^{T(p_1)}[i+1,i_0-1,j+1,j_0-1] + L^p[i_0+1,i',j_0+1,j'] + \alpha(p_1,p_2) & \end{cases}$$

Fig. 5. The recursions for the table L^p.

the right instead of to the left. If hchild(p) is defined, we initialize all entries with $i' \le \min(\text{hchild}(p)^{\mathrm{R}} + d, p^{\mathrm{R}})$ as $R^p[i,i',j,j'] = L^p[i,i',j,j']$. All other items are computed according to the recursion shown in Fig. 6. If hchild(p) is not defined, we initialize all items with $i' = p^{\mathrm{L}} + d$ as $R^p[i,i',j,j'] = M[i,i',j,j']$. The recursion for R^p in this case includes lines I, II, III, and V from Fig. 6.

Once the tables are computed, the actual alignment can be constructed using the usual backtrace technique.

4.3 Correctness

Let (A,P) be an optimal alignment and consensus structure for the fragments corresponding to some table entry $M[i,i',j,j']$, $L^p[i',i,j',j]$, or $R^p[i,i',j,j']$ (note the swapped indices in the entry of L^p). In all recursions, lines I and II cover the cases where A aligns i' or j' to a gap. Line III covers the cases where $(i',j') \in A$ and no arcs of P are adjacent to i' or j'. Furthermore i' and j' can never be adjacent to arcs of the consensus structure whose other end is outside of the current fragment (due to the semantics of the table entries). Hence, the case that remains is where i' and j' are one end of some arc of the consensus structure whose other end is also contained in the current fragment. In the recursion for M, this case is covered in line IV, and in the recursions for L and R this case is further decomposed into subcases corresponding to lines IV to VI. In

$$R^p[i,i',j,j'] = \min \begin{cases} R^p[i,i'-1,j,j'] + \gamma_1(i) & \text{I} \\ R^p[i,i',j,j'-1] + \gamma_2(j) & \text{II} \\ R^p[i,i'-1,j,j'-1] + \beta(i,j) & \text{III} \\ \text{for all } p_1 = (i_0,i') \in P_1,\ p_2 = (j_0,j') \in P_2 \text{ with } i \le i_0,\ j \le j_0, & \\ \text{and hchild}(p) \text{ is nested in } p_1 & \text{IV} \\ \quad M[i,i_0-1,j,j_0-1] + R^p[i_0+1,i'-1,j_0+1,j'-1] + \alpha(p_1,p_2) & \\ \text{for all } p_1 = (i_0,i') \in P_1,\ p_2 = (j_0,j') \in P_2 \text{ with } i \le i_0,\ j \le j_0, & \\ \text{hchild}(p) \text{ is not nested in } p_1, \text{ and span}(p_1) \le 2d & \text{V} \\ \quad R^p[i,i_0-1,j,j_0-1] + M[i_0+1,i'-1,j_0+1,j'-1] + \alpha(p_1,p_2) & \\ \text{for all } p_1 = (i_0,i') \in P_1,\ p_2 = (j_0,j') \in P_2 \text{ with } i \le i_0,\ j \le j_0, & \\ \text{hchild}(p) \text{ is not nested in } p_1, \text{ and span}(p_1) > 2d & \text{VI} \\ \quad R^p[i,i_0-1,j,j_0-1] + R^{T(p_1)}[i_0+1,i'-1,j_0+1,j'-1] + \alpha(p_1,p_2) & \end{cases}$$

Fig. 6. The recursions for the table R^p.

all those cases, the fragment is decomposed in the arc match (p_1, p_2), the fragment below the arc match and the fragment before it (or behind it, in the case of the table L). This decomposition is correct since the consensus structure is nested and hence cannot contain other arc pairs whose arcs cross p_1 and p_2 to connect the fragments before and below (p_1, p_2). It remains to show that in each case the table entries we recursively descend to exist.

Fix an arc $p \in P_T$ for which $\text{hchild}(p)$ is defined (the case where $\text{hchild}(p)$ is not defined is similar). Let $p_1 = (i_0, i')$ be an arc considered in lines IV to VI of the recursion for R^p.

Lemma 5. *p_1 does not cross* $\text{hchild}(p)$.

Proof. Since the case $i' \le \min(\text{hchild}(p)^{\text{R}} + d, p^{\text{R}})$ is handled by the initialization of R^p, we have $i' > \min(\text{hchild}(p)^{\text{R}} + d, p^{\text{R}})$. Therefore, either $i' > \text{hchild}(p)^{\text{R}} + d$ or $i' > p^{\text{R}}$. In the former case we have from the assumption that (S_1, P_1) is d-crossing that p_1 does not cross $\text{hchild}(p)$. In the latter case we also have that p_1 does not cross $\text{hchild}(p)$ since otherwise, p_1 would also cross p, contradicting Lemma 1. □

By Lemma 5, either $\text{hchild}(p)$ is nested in p_1 or $\text{hchild}(p)$ precedes p_1. The case where $\text{hchild}(p)$ is nested in p_1 is handled in line VI of the recursion. In this case we have that either $T(p_1) = p$ or p is nested in p_1. In both cases we have that $i_0 \le p^{\text{L}} + d - 1$ (due to Lemma 3). From this inequality we obtain that $(i_0 - 1) - i = (i_0 - p^{\mathrm{L}}) + (p^{\mathrm{L}} - i) - 1 \leq 2d - 2$, so the entry $M[i, i_0 - 1, j, j_0 - 1]$ exists. Moreover, from the inequality $i_0 \leq p^{\mathrm{L}} + d - 1$ and the assumption that $\mathrm{hchild}(p)$ is nested in p_1 we obtain that the entry $R^p[i_0 + 1, i' - 1, j_0 + 1, j' - 1]$ exists.

Now consider the case where $\mathrm{hchild}(p)$ precedes p_1 which is handled in lines V and VI of the recursion. In both lines, the common entry $R^p[i, i_0 - 1, j, j_0 - 1]$ exists.

If $\mathrm{span}(p_1) \leq 2d$ then the entry $M[i_0 + 1, i' - 1, j_0 + 1, j' - 1]$ exists since $(i' - 1) - (i_0 + 1) = \mathrm{span}(p_1) - 3 \leq 2d - 3$. If $\mathrm{span}(p_1) > 2d$ then we need to show that the entry $R^{T(p_1)}[i_0, i', j_0, j']$ exists. We have that $p_1^{\mathrm{R}} - p^{\mathrm{R}} > \mathrm{span}(\mathrm{hchild}(p)) > 2d$, and therefore p_1 does not cross p and $p_1 \notin I_d(p)$. It follows that $T(p_1) \neq p$. Therefore, $T(p_1)$ is nested in p, so the table $R^{T(p_1)}$ was already filled by the algorithm when the table R^p is filled. From Lemma 3 and Lemma 4 we conclude that the entry $R^{T(p_1)}[i_0, i', j_0, j']$ exists. The correctness arguments for the recursion for L^p are analogous.

4.4 Time Complexity

Let $d_k^R(i)$ (resp., $d_k^L(i)$) denote the number of arcs p in P_k with $p^{\mathrm{R}} = i$ (resp., $p^{\mathrm{L}} = i$). Let $d_k(i) = d_k^R(i) + d_k^L(i) + 1$. In stage 1, the time complexity for computing an entry $M[i, i', j, j']$ is $O((1 + d_1^R(i'))(1 + d_2^R(j'))) = O(d_1(i')d_2(j'))$. For fixed i' and j', the number of entries of the form $M[i, i', j, j']$ that are computed by the algorithm is $O(dm)$. Therefore, the time complexity of stage 1 is $O\left(\sum_{i'=1}^{n} \sum_{j'=1}^{m} dm \cdot d_1(i')d_2(j')\right) = O\left(dm \sum_{i'=1}^{n} d_1(i') \sum_{j'=1}^{m} d_2(j')\right) = O(dnm^2)$.

For $p \in P_T$, the time complexity of computing an entry $L^p[i, i', j, j']$ is $O((1 + d_1^L(i))(1 + d_2^L(j))) = O(d_1(i)d_2(j))$, and the time complexity of computing an entry $R^p[i, i', j, j']$ is $O((1 + d_1^R(i'))(1 + d_2^R(j'))) = O(d_1(i')d_2(j'))$. Consider some fixed arc p and fixed indices i and j. Let $c_{i,j}^p$ denote the number of computed entries of the form $L^p[i, i', j, j']$ or $R^p[i', i, j', j]$. Then stage 2 requires $O\left(\sum_{p \in P_T} \sum_{i=1}^{n} \sum_{j=1}^{m} c_{i,j}^p \cdot d_1(i)d_2(j)\right)$ time.

For every $p \in P_T$, $c_{i,j}^p \in O(dm)$ for all i and j. Assuming i and j are fixed, we now count the number of arcs $p \in P_T$ for which $c_{i,j}^p > 0$. Let p_0 be the minimum span tree arc such that $i \in [p_0^{\mathrm{L}}, p_0^{\mathrm{R}}]$. If p is a tree arc with $c_{i,j}^p > 0$ then p satisfies one of the following:

1. p is nested in p_0, $p^{\mathrm{R}} < i$, and p^{R} is maximal among all tree arcs that satisfy the previous two conditions.

2. p is nested in p_0, $p^{\mathrm{L}} > i$, and p^{L} is minimal among all tree arcs that satisfy the previous two conditions.
3. p_0 is nested in p and $i \notin [\mathrm{hchild}(p)^{\mathrm{L}}, \mathrm{hchild}(p)^{\mathrm{R}}]$.

There are at most two arcs of types 1 and 2 above. Let $p_0, p_1, \ldots, p_k$ be all the tree arcs of the third type, such that p_i is nested in p_{i+1} for all i. Since $\mathrm{span}(p_i) \leq \mathrm{span}(\mathrm{hchild}(p_{i+1}))$, we have $\mathrm{span}(p_{i+1}) > 2 \cdot \mathrm{span}(p_i)$ for all i and therefore $k < \log_2 n$. Thus, the time complexity of stage 2 is $O\left(\sum_{i=1}^{n} \sum_{j=1}^{m} dm \log n \cdot d_1(i) d_2(j)\right) = O(dm^2 n \log n)$.

5 Conclusion

We presented an algorithm that computes the optimal sequence structure alignment for a nested consensus structure and crossing input structures. In practice, crossing input structures can be used to represent several sub-optimal structures simultaneously, from which the alignment effectively selects the most appropriate one. On the theoretical side, we generalized the optimizations developed by Klein [14] to crossing input structures. In future work, we will try to incorporate also the space optimization and the optimization of Demaine et al [2].

Acknowledgements. This work is partially funded by DFG grant BA 2168/3-1, the Israel Science Foundation grant 35/05 and the Israel-Korea Scientific Research Cooperation.

References

1. Jiang, T., Lin, G., Ma, B., Zhang, K.: A general edit distance between RNA structures. J. Comput. Biol. **9**(2) (2002) 371–88
2. Demaine, E.D., Mozes, S., Rossman, B., Weimann, O.: An optimal decomposition algorithm for tree edit distance. In Arge, L., Cachin, C., Jurdzinski, T., Tarlecki, A., eds.: Proc. of 34th International Colloquium on Automata, Languages and Programming. Volume 4596 of Lecture Notes in Computer Science., Springer (2007) 146–157
3. Washietl, S., Hofacker, I.L., Lukasser, M., Huttenhofer, A., Stadler, P.F.: Mapping of conserved RNA secondary structures predicts thousands of functional noncoding RNAs in the human genome. Nat Biotechnol **23**(11) (2005) 1383–90
4. Consortium, A.F.B., Backofen, R., Bernhart, S.H., Flamm, C., Fried, C., Fritzsch, G., Hackermuller, J., Hertel, J., Hofacker, I.L., Missal, K., Mosig, A., Prohaska, S.J., Rose, D., Stadler, P.F., Tanzer, A., Washietl, S., Will, S.: RNAs everywhere: genome-wide annotation of structured RNAs. J Exp Zoolog B Mol Dev Evol **308**(1) (2007) 1–25
5. Waterman, M., Smith, T.: RNA secondary structure: a complete mathematical analysis. Math. Biosci. **42** (1978) 257–266

6. Nussinov, R., Jacobson, A.: Fast algorithm for predicting the secondary structure of single-stranded RNA. Proc. Natl. Acad. Sci. **77**(11) (1980) 6309–6313
7. Zuker, M., Stiegler, P.: Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. Nucleic Acids Research **9**(1) (1981) 133–148
8. Akutsu, T.: Approximation and exact algorithms for RNA secondary structure prediction and recognition of stochastic context-free languages. Journal of Combinatorial Optimization **3** (1999) 321336
9. Wexler, Y., Zilberstein, C., Ziv-Ukelson, M.: A study of accessible motifs and RNA folding complexity. Journal of Computational Biology **14**(6) (2007) 856–872
10. Do, C.B., Woods, D.A., Batzoglou, S.: CONTRAfold: RNA secondary structure prediction without physics-based models. Bioinformatics **22**(14) (2006) e90–8
11. Gardner, P.P., Giegerich, R.: A comprehensive comparison of comparative RNA structure prediction approaches. BMC Bioinformatics **5** (2004) 140
12. Hofacker, I.L., Fekete, M., Stadler, P.F.: Secondary structure prediction for aligned RNA sequences. Journal of Molecular Biology **319**(5) (2002) 1059–66
13. Seemann, S.E., Gorodkin, J., Backofen, R.: Unifying evolutionary and thermodynamic information for RNA folding of multiple alignments. Nucleic Acids Research (2008)
14. Klein, P.: Computing the edit-distance between unrooted ordered trees. In Bilardi, G., Italiano, G.F., Pietracaprina, A., cci, G.P., eds.: Proceedings of the 6th Annual European Symposium, Venice, Italy, Springer-Verlag, Berlin (1998) 91–102
15. Sankoff, D.: Simultaneous solution of the RNA folding, alignment and protosequence problems. SIAM J. Appl. Math. **45**(5) (1985) 810–825
16. Mathews, D.H., Turner, D.H.: Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. Journal of Molecular Biology **317**(2) (2002) 191–203
17. Havgaard, J.H., Lyngso, R.B., Stormo, G.D., Gorodkin, J.: Pairwise local structural alignment of RNA sequences with sequence similarity less than 40 Bioinformatics **21**(9) (2005) 1815–24
18. M. Ziv-Ukelson, I. Gat-Viks, Y.W., Shamir, R.: A faster algorithm for RNA cofolding. Proc. Algorithms in Bioinformatics, 8th International Workshop, WABI, LNBI 5251 (September 2008) 174–185
19. Will, S., Reiche, K., Hofacker, I.L., Stadler, P.F., Backofen, R.: Inferring noncoding RNA families and classes by means of genome-scale structure-based clustering. PLOS Computational Biology **3**(4) (2007) e65
20. Evans, P.A.: Finding common rna pseudoknot structures in polynomial time. In: Combinatorial Pattern Matching (CPM 2006). Volume 4009/2006 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2006) 223–232
21. Evans, P.A.: Algorithms and Complexity for Annotated Sequence Analysis. PhD thesis, University of Alberta (1999)