# Multiple RNA Interaction with Sub-optimal Solutions

Syed Ali Ahmed* and Saad Mneimneh**

The Graduate Center and Hunter College, City University of New York, New York, USA
sahmed3@gc.cuny.edu, saad@hunter.cuny.edu

**Abstract.** The interaction of two RNA molecules involves a complex interplay between folding and binding that warranted recent developments in RNA-RNA interaction algorithms. However, biological mechanisms in which more than two RNAs take part in an interaction exist. It is reasonable to believe that interactions involving multiple RNAs are generally more complex to be treated pairwise. In addition, given a pool of RNAs, it is not trivial to predict which RNAs are interacting without sufficient biological knowledge. Therefore, structures resulting from multiple RNA interactions often cannot be predicted by the existing algorithms.

We recently proposed a system for multiple RNA interaction that overcomes the difficulties mentioned above by formulating a combinatorial optimization problem called *Pegs and Rubber Bands*. A solution to this problem encodes a structure of interacting RNAs. In general, however, the optimal solution obtained does not necessarily correspond to the actual structure observed experimentally. Moreover, a structure produced by interacting RNAs may not be unique. In this work, we extend our previous approach to generate multiple sub-optimal solutions. By clustering these solutions, we are able to reveal representatives that correspond to realistic structures. Specifically, our results on the U2-U6 complex in the spliceosome of yeast and the CopA-CopT complex in E. Coli are consistent with published biological structures.

## 1 Introduction

The interaction of two RNA molecules has been independently formulated as a computational problem in several works, e.g. [1–3]. In their most general form, these formulations lead to NP-hard problems (which means computationaly intractable, i.e. the running time of the algorithm that produces an optimal solution increases exponentially with the problem size). To overcome this hurdle, researchers have been either reverting to approximation algorithms, or imposing algorithmic restrictions; for instance, the avoidance of the formation of certain structures.

* Supported by NSF Award CCF-AF 1049902 and a CUNY GC Science Fellowship.
** Corresponding author. Supported by NSF Award CCF-AF 1049902.

M. Basu, Y. Pan, and J. Wang (Eds.): ISBRA 2014, LNBI 8492, pp. 149–162, 2014.

While these algorithms had limited use in the beginning, they became important venues for (and in fact popularized) an interesting biological fact: RNAs interact. For instance, micro-RNAs (miRNAs) bind to a complementary part of messenger RNAs (mRNAs) and inhibit their translation [4]. But more complex forms of RNA-RNA interaction exist. In E. Coli, CopA binds to the ribosome binding site of CopT, also as a regulation mechanism to prevent translation [5]; so does OxyS to fhlA [6]. In both of these structures, the simultaneous folding (within the RNA) and binding (to the other RNA) are non-trivial to be predicted as separate events. To account for this, most of the RNA-RNA interaction algorithms calculate the probability for a pair of subsequences (one of each RNA) to participate in the interaction, and in doing so they generalize the energy model used for the partition function of a single RNA to the case of two RNAs [7–12]. This generalization takes into consideration the simultaneous aspect of folding and binding.

Not surprisingly, there exist other mechanisms in which more than two RNA molecules take part in an interaction. Typical scenarios involve the interaction of multiple small nucleolar RNAs (snoRNAs) with ribosomal RNAs (rRNAs) in guiding the methylation of the rRNAs [4], and multiple small nuclear RNAs (snRNA) with mRNAs in the splicing of introns [13]. Even with the existence of a computational framework for a single RNA-RNA interaction, it is reasonable to believe that interactions involving multiple RNAs are generally more complex to be treated pairwise. In addition, given a pool of RNAs, it is not trivial to predict which RNAs interact without some prior biological information. Some attempts for multiple RNA interaction have been considered, e.g. [14, 15], but they only generalize the partition function algorithm of [16] by concatenation of all RNAs into one, and so can only produce restricted structures, e.g. no kissing loops. Even though algorithms for kissing loops exist, e.g. [17], advances in pairwise interaction of RNAs suggest that the concatenation model is less suitable.

We recently proposed a new computational approach for handling multiple RNA interaction based on a combinatorial optimization problem that we call Pegs and Rubber Bands [18]. In this work, we extend this approach to generate multiple sub-optimal solutions, and show that these solutions correspond to realistic structure.

## 2 Background and Approach

### 2.1 Pegs and Rubber Bands: A Formulation

We now present the problem of Pegs and Rubber Bands as a framework for multiple RNA interaction. The link between the two will be made shortly following a formal description of Pegs and Rubber Bands.

Consider $m$ levels numbered 1 to $m$ with $n_l$ pegs in level $l$ numbered 1 to $n_l$. There is an infinite supply of rubber bands that can be placed around two pegs in consecutive levels. For instance, we can choose to place a rubber band around peg $i$ in level $l$ and peg $j$ in level $l+1$; we call it a rubber band at $[l, i, j]$. Every such pair of pegs $[l, i]$ and $[l+1, j]$ contribute their own weight $w(l, i, j)$. The Pegs

and Rubber Bands problem is to maximize the total weight by placing rubber bands around pegs in such a way that no two rubber bands intersect. In other words, each peg can have at most one rubber band around it, and if a rubber band is placed at $[l, i_1, j_1]$ and another at $[l, i_2, j_2]$, then $i_1 < i_2 \Leftrightarrow j_1 < j_2$. We assume without loss of generality that $w(l, i, j) \neq 0$ to avoid the unnecessary

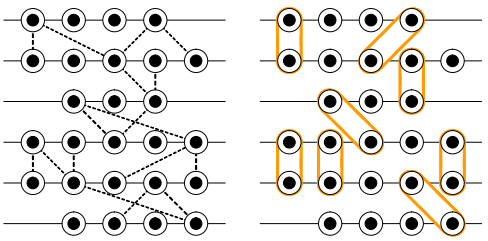

**Fig. 1.** Pegs and Rubber Bands. All positive weights are equal to 1 and are represented by dashed lines. The optimal solution achieves a total weight of 8.

placement of rubber bands and, therefore, either $w(l, i, j) > 0$ or $w(l, i, j) = -\infty$. Figure 1 shows an example.

Given an optimal solution, it can always be reconstructed from left to right by repeatedly placing some rubber band at $[l, i, j]$ such that, at the time of this placement, no rubber band is around peg $[l, k]$ for $k > i$ and no rubber band is around peg $[l + 1, k]$ for $k > j$. This process can be carried out by a dynamic programming algorithm to compute the maximum weight (Section 3.1).

### 2.2 Multiple RNA Interaction as Pegs and Rubber Bands

To provide some initial context we now describe how the formulation of Pegs and Rubber Bands, though in a primitive way, captures the problem of multiple RNA interaction. We think of each level as an RNA and each peg as one base of the RNA. The weight $w(l, i, j)$ corresponds to the negative of the energy contributed by the binding of the $i^{\text{th}}$ base of RNA $l$ to the $j^{\text{th}}$ base of RNA $l + 1$. This can be obtained using existing algorithms for RNA-RNA interaction that act on pairs of RNAs. It should be clear, therefore, that an optimal solution for Pegs and Rubber Bands represents the lowest energy conformation in a base-pair energy model, when a pseudoknot-like restriction is imposed on the RNA interaction (rubber bands cannot intersect). In doing so, we obviously assume that an order on the RNAs is given with alternating sense and antisense, and that the first RNA interacts with the second RNA, which in turn interacts with the third RNA, and so on. We later relax this ordering and the stringency of the interaction pattern of the RNAs. While a simple base-pairing model is not likely to give realistic results, our goal here was simply to establish a correspondence between the two problems.

### 2.3 Windows and Gaps: A Better Formulation for RNA Interaction

In the previous section, we described our initial attempt to view the interaction of $m$ RNAs as a Pegs and Rubber Bands problem with $m$ levels, where the first RNA interacts with the second RNA, and the second with the third, and so on (so they alternate in sense and antisense). This used a simple base-pair energy model, which is not too realistic. We now address this issue (and leave the issues of the ordering and the interaction pattern to the following section). A better model for RNA interaction will consider windows of interaction instead of single bases. For instance, subsequence $[i_1, i_2]$ of RNA $l$ can interact with subsequence $[j_1, j_2]$ of RNA $l+1$. In terms of our Pegs and Rubber Bands problem, this translates to placing rubber bands around a stretch of contiguous pegs in two consecutive levels, e.g. around pegs $[l, i_1]$, $[l, i_2]$, $[l+1, j_1]$, and $[l+1, j_2]$, where $i_2 \geq i_1$ and $j_2 \geq j_1$. The weight contribution of placing such a rubber band is now given by $w(l, i_2, j_2, u, v)$, where $i_2$ and $j_2$ are the last two pegs covered by the rubber band in level $l$ and level $l+1$, and $u = i_2 - i_1 + 1$ and $v = j_2 - j_1 + 1$ represent the length of the two windows covered in level $l$ and level $l+1$, respectively.

As a *heuristic*, we also allow for the possibility of imposing a gap $g \geq 0$ between windows as a way to ensure that windows are energetically independent. This gap is also taken into consideration when we perform the gap filling procedure described in Section 3.1.

We use windows satisfying $2 \leq u, v \leq w = 26$ and a gap $g = 0$. The weights $w(l, i, j, u, v)$ are obtained from RNAup, a tool to compute energies of pairwise interactions [7], as (negative of energy values):

$$w(l, i, j, u, v) \propto \log p_l(i - u + 1, i) + \log p_{l+1}(j - v + 1, j)$$
$$+ \log Z_l^I(i - u + 1, i, j - v + 1, j)$$

where $p_l(i_1, i_2)$ is the probability that subsequence $[i_1, i_2]$ is free (does not fold) in RNA $l$, and $Z_l^I(i_1, i_2, j_1, j_2)$ is the partition function (as computed in [7]) of the interaction of subsequences $[i_1, i_2]$ in RNA $l$ and $[j_1, j_2]$ in RNA $l+1$ (subject to no folding within RNAs). As such, the weight considers intra-molecular and inter-molecular energies. The windows are filtered for sub-additivity as described in Section 3.1.

### 2.4 Order and Interaction Pattern via Permutations

We now describe how to relax the ordering and the stringency of the interaction pattern of the RNAs. We first identify each RNA as being *even* (sense) or *odd* (antisense), but this convention can obviously be switched. Given $m$ RNAs and a permutation on the set $\{1, \ldots, m\}$, we map the RNAs onto the levels of a Pegs and Rubber Bands problem as follows: We place the RNAs in the order in which they appear in the permutation on the same level as long as they have the same parity (they are either all even or all odd). We then increase the number of levels by one, and repeat. RNAs that end up on the same level are *virtually* considered as one RNA that is the concatenation of all. However, in

the corresponding Pegs and Rubber Bands problem, we do not allow windows to span multiple RNAs, nor do we enforce a gap between two windows in different RNAs. We describe in Section 3.2 a greedy algorithm that searches heuristically for the best permutation.

## 3 Algorithms

### 3.1 Complexity of the Problem and Approximations

We proved that Pegs and Rubber Bands is NP-hard [18]. Therefore, any algorithm that finds an optimal solution generally requires exponential time. However, while our problem is NP-hard, we also proved that the same formulation can be adapted to obtain a polynomial time approximation. A maximization problem admits a polynomial time approximation scheme (PTAS) iff for every **fixed** $\epsilon > 0$ there is an algorithm with a running time polynomial in the size of the input that finds a solution within $(1 - \epsilon)$ of optimal [19].

Let $OPT$ be the weight of the optimal solution and denote by $W[i \dots j]$ the weight of the optimal solution when the problem is restricted to levels $i, i + 1, \dots, j$ (a sub-problem). For a given $\epsilon > 0$, let $k = \lceil \frac{1}{\epsilon} \rceil$. Consider the following $k$ solutions (weights), each obtained by a concatenation of optimal solutions for sub-problems consisting of at most $k$ levels.

$$
\begin{aligned}
W_1 &= W[1 \dots 1] + W[2 \dots k+1] + W[k+2 \dots 2k+1] + \dots \\
W_2 &= W[1 \dots 2] + W[3 \dots k+2] + W[k+3 \dots 2k+2] + \dots \\
&\vdots \\
W_k &= W[1 \dots k] + W[k+1 \dots 2k] + W[2k+1 \dots 3k] + \dots
\end{aligned}
$$

The best of these solutions is a $(1 - \epsilon)$ approximation [18], i.e.

$$\max_i W_i \geq \frac{k-1}{k} OPT \geq (1 - \epsilon) OPT$$

Therefore, for a given integer $k$, the $(1 - 1/k)$-factor approximation algorithm is to simply choose the best $W_i = W[1 \dots i] + W[i+1 \dots i+k] + W[i+k+1 \dots i+2k] + \dots$ as a solution, where $W[i \dots j]$ denotes the weight of the optimal solution for the sub-problem consisting of levels $i, i+1 \dots, j$. Some more theoretical results on approximation based on our formulation were obtained in [20].

As a practical step, and instead of using the $W_i$'s for the comparison, we can fill in for each $W_i$ some additional rubber bands (interactions) between (RNAs) level $i$ and level $i + 1$, between level $i + k$ and level $i + k + 1$, and so on, by identifying the pegs of these levels (regions of RNAs) that are not part of the solution. This does not affect the theoretical guarantee but gives a larger weight to the solution. We call it *gap filling*.

Figure 2 describes an algorithm for $m$ levels based on dynamic programming by defining $W(i_1, i_2, \dots, i_m)$ to be the maximum weight when we truncate

the levels at pegs $[1, i_1], [2, i_2], \ldots, [m, i_m]$. The maximum weight is given by $W(n_1, n_2, \ldots, n_m)$ and the optimal solution can be obtained by standard backtracking.

$$W(i_1, i_2, \ldots, i_m) = \max \begin{cases} W(i_1 - 1, i_2, \ldots, i_m) \\ W(i_1, i_2 - 1, i_3, \ldots, i_m) \\ \vdots \\ W(i_1, \ldots, i_{m-1}, i_m - 1) \\ W((i_1 - u - g)^+, (i_2 - v - g)^+, i_3, \ldots, i_m) + w(1, i_1, i_2, u, v) \\ W(i_1, (i_2 - u - g)^+, (i_3 - v - g)^+, i_4, \ldots, i_m) + w(2, i_2, i_3, u, v) \\ \vdots \\ W(i_1, \ldots, i_{m-2}, (i_{m-1} - u - g)^+, (i_m - v - g)^+) + w(m - 1, i_{m-1}, i_m, u, v) \end{cases}$$

where $x^+$ denotes $\max(0, x)$, $w(l, i, j, u, v) = -\infty$ if $u > i$ or $v > j$, $0 < u, v \leq w$ (the maximum window size), $g \geq 0$ (the gap), and $W(0, 0, \ldots, 0) = 0$.

**Fig. 2.** Dynamic programming algorithm for Pegs and Rubber Bands with the windows and gaps formulation

The running time of the algorithm is $O(mw^2 n^m)$ (exponential) and $O(mw^2 \lceil \frac{1}{\epsilon} \rceil n^{\lceil \frac{1}{\epsilon} \rceil})$ for the approximation scheme (polynomial), where $w$ is the maximum window length. If we impose that $u = v$ in $w(l, i, j, u, v)$, then those running times become $O(mwn^m)$ and $O(mw \lceil \frac{1}{\epsilon} \rceil n^{\lceil \frac{1}{\epsilon} \rceil})$ respectively.

For the correctness of the algorithm, we have to assume that windows are *sub-additive*. In other words, we require the following condition (otherwise, the algorithm may compute an incorrect optimum due to the possibility of achieving the same window by two or more smaller ones with higher total weight):

$$w(l, i, j, u_1, v_1) + w(l, i - u_1, j - v_1, u_2, v_2)$$
$$\leq w(l, i, j, u_1 + u_2, v_1 + v_2)$$

In our experience, most existing RNA-RNA interaction algorithms produce weights (the negative of the energy values) of RNA interaction windows that mostly conform to the above condition. In rare cases, we filter the windows to eliminate those that are not sub-additive. For instance, if the above condition is not met, we set $w(l, i, j, u_1, v_1) = w(l, i - u_1, j - v_1, u_2, v_2) = -\infty$.

### 3.2 Heuristic for a Single Solution

A heuristic for resolving the ordering and the interaction pattern of the RNAs is shown in Figure 3. As described in Section 2.4, the order and interaction pattern are determined by a permutation. The main idea of this heuristic is to first start with an arbitrary permutation, and then iteratively change it by moving along neighboring permutations with better solutions (larger weights). This is repeated until no more improvement can be achieved. Using the PTAS, this algorithm finds **one** solution within a $(1 - \epsilon)$-factor of optimal (which could itself be the optimal).

```
Given ϵ = 1/k and m RNAs
    produce a random permutation π on {1, ..., m}
    let W be the weight of the (1 − ϵ)-optimal solution given π
    repeat
      better←false
      generate a set Π of neighboring permutations for π
      for every π′ ∈ Π (in any order)
          let W′ be the weight of the (1 − ϵ)-optimal solution given π′
          if W′ > W
            then W ← W′
                 π ← π′
                 better←true
    until not better
```

**Fig. 3.** A heuristic for multiple RNA interaction using the PTAS algorithm

To generate neighboring permutations for this heuristic algorithm one could adapt a standard 2-opt method used in the Traveling Salesman Problem (or other techniques). For instance, given permutation $\pi$, a neighboring permutation $\pi'$ can be obtained by dividing $\pi$ into three parts and making $\pi'$ the concatenation of the first part, the reverse of the second part, and the third part. In other words, if $\pi = (\alpha, \beta, \gamma)$, then $\pi' = (\alpha, \beta^R, \gamma)$ is a neighbor of $\pi$, where $\beta^R$ is the reverse of $\beta$.

### 3.3 Multiple Sub-optimal Solutions

We now describe how to generate (all) solutions with a weight of at least some threshold $T$.

**Generation:** RNAs often interact in more than one way. To explore this, we assume that the order and interaction pattern have been already determined, e.g. by the algorithm of Section 3.2. We then seek sub-optimal solutions. Denote by $S(i_1, \ldots, i_m)$ a solution where $i_l$ is the smallest index at level $l$ such that peg $[l, i_l]$ is covered by a window, $l = 1 \ldots m$. We will also use $S(i_1, \ldots, i_m)$ interchangeably to represent the weight of that solution. Similarly, we will use $w(l, i, j, u, v)$ interchangeably to denote a window and its weight. We denote by $S(i_1, \ldots, i_m)+w(l, i, j, u, v)$ an extension of solution $S$ by the addition of window $w$.

We say that a window $w(l, i, j, u, v)$ in $S(i_1, \ldots, i_m)$ is a *terminal* window iff:

- $i - u + 1 = i_l$,
- $j - v + 1 = i_{l+1}$, and
- no other window $w(l', i', j', u', v')$ in $S(i_1, \ldots, i_m)$ satisfies $i' - u' + 1 = i_{l'}$, $j' - v' + 1 = i_{l'+1}$, and $l' > l$.

This imposes some order on the windows to prevent generating the same solution in multiple ways. To that end, we can only extend a solution by adding to it a terminal window (a window that becomes the terminal for the extended solution). Observe that whenever $W(i_1-g-1, \ldots, i_m-g-1)+S(i_1, \ldots, i_m) < T$,

where $g$ is the gap parameter as described in Section 2.3, $S$ cannot be extended in anyway to meet the threshold.

Let $\phi = S(n_1 + g + 1, \ldots, n_m + g + 1)$ represent the empty solution (with zero weight). We have the following algorithm (Figure 4) for generating every solution with weight at least $T$, starting with Process($\phi$). Because windows are considered in order, the running time of the algorithm is linear in the size of its output plus a crude $O(2^{|\mathbb{W}|})$ bound (all possible solutions), where $\mathbb{W}$ is the set of windows.

```
Process(S(i_1, ..., i_m))
  if W(i_1 - g - 1, ..., i_m - g - 1) + S(i_1, ..., i_m) < T
    then return
    else for every window w(l, i, j, u, v) that is
              terminal in S(i_1, ..., i_m) + w(l, i, j, u, v)
              with i_l - i > g and i_{l+1} - j > g
              Process(S(i_1, ..., i_m) + w(l, i, j, u, v))
  if S(i_1, ..., i_m) ≥ T
    then output S
```

**Fig. 4.** Generating multiple sub-optimal solutions

**Clustering:** The sub-optimal solutions generated above may be a lot more than what we need. We use a pseudo-clustering algorithm to identify a small set of representative solutions. We use the term pseudo-clustering because our algorithm does not attempt to optimize clusters in any way. Let $d(S, C)$ be the distance between a solution $S$ and a cluster $C$ (Section 3.3.3), and assume a threshold $D$. The idea is to add a solution $S$ to a cluster $C$ if $d(S, C) < D$. Figure 5 shows an algorithm that clusters solutions until all solutions are in clusters or the maximum number of clusters $c$ has been reached. The running time of this algorithm is, therefore, $O(|\mathbb{S}|cf(m, n))$, where $\mathbb{S}$ is the set of solutions, and $f(m, n)$ is the time needed to compute the distance on instances with $m$ RNAs of length $n$.

```
Cluster
  r = 0
  for every solution S in decreasing order of weight
      if there exists a cluster C_i such that d(S, C_i) < D
        then C_i ← C_i ∪ {S}
        else r ← r + 1
             C_r ← {S}
             output S (the best in its cluster)
             if r =maximum number of clusters c
               return
```

**Fig. 5.** An algorithm for pseudo-clustering the solutions

**Distance.** Recall that peg $[l, i]$ represents the $i^{\text{th}}$ base of RNA $l$. Therefore, if peg $[l, i]$ is covered by a window in some solution for Pegs and Rubber Bands, we say that base $i$ is interacting. Otherwise, we distinguish between two cases: base $i$ is free, or there is a base $j$ of RNA $l$ that is **not** interacting such that base

$i$ folds onto base $j$ (makes a bond) in the optimal folding of RNA $l$. Therefore, given a solution, RNA $l$ can be represented by a string $s_l$ where $s_l[i]$, the $i^{\text{th}}$ character in $s_l$, is one of three letters: I for interacting (with another RNA), F for free, and B for bonding (to the same RNA).

We define a distance function $d(S_1, S_2)$ between two solutions, and set $d(S, C)$ as the distance between $S$ and the representative solution of cluster $C$. The last paragraph in this section describes how such a representative is determined.

Jaccard: Given a solution $S$, convert $s_l$ for every $l = 1 \dots m$ into a binary vector $v_l$ by replacing I with 10, B with 01, and F with 00. Concatenate all such vectors into one vector $v = v_1 v_2 \dots v_m$. If $u$ is the vector corresponding to solution $S_1$ and $v$ is the vector corresponding to solution $S_2$, then:

$$d(S_1, S_2) = \frac{\sum_i u[i] \otimes v[i]}{\sum_i u[i] \oplus v[i]}$$

where $v[i]$ is the $i^{\text{th}}$ bit of vector $v$, and $\otimes$ and $\oplus$ stand for the binary operators XOR (exclusive OR) and OR, respectively. Intuitively, this reflects a Hamming distance scaled by the number of entries that can potentially differ [21]. We also define a coarser version of this distance below.

Levenshtein: Given a solution $S$, collapse $s_l$ for every $l = 1 \dots m$ by replacing repeated consecutive letters by one occurrence of the given letter, e.g. replace BBBBB by B. With this modification, if $s_1, \dots, s_m$ correspond to solution $S_1$ and $t_1, \dots, t_m$ correspond to solution $S_2$, then:

$$d(S_1, S_2) = \frac{\sum_{l=1}^{m} \text{Lev}(s_l, t_l)}{\sum_{l=1}^{m} \max(|s_l|, |t_l|)}$$

where $\text{Lev}(s, t)$ is the Levenshtein distance (in modern terms, an edit distance where each mismatch and deletion contributes a 1 [22]), and $|\ \ |$ denotes the length of a string.

We either use the Jaccard distance, or the average of Jaccard and Levenshtein when the Jaccard distance is not sensitive to small variations. In computing $d(S, C)$, the representative of cluster $C$ is either the best solution in the cluster (i.e. the one with the largest weight, which is also the one that started the cluster), or the consensus of the cluster. The consensus can be obtained in terms of the vector $v$, where $v[i] = 1$ for the consensus solution if and only if a strict majority of the solutions in $C$ have the $i^{\text{th}}$ bit equal to 1.

## 4 Results

For all of our experiments, we only show the interaction pattern (no folding within the individual RNAs).

### 4.1 Single Solutions

We use the algorithm for Section 3.2 We pick the largest weight solution among several runs of the algorithm. The value of $k$ and the gap filling criterion depend on the scenario, as described below.

**Fishing for Pairs.** Six RNAs in E. Coli of which three pairs are known to interact are used [8]. The interest here is to see whether the algorithm can identify the three pairs. For this purpose, it will suffice to set $k = 2$ and to ignore gap filling. Furthermore, we only consider solutions in which each RNA interacts with at most one other RNA. The solution with the largest weight identifies the three pairs correctly (Figure 6). In addition, the interacting sites in each pair are consistent with the predictions of existing RNA-RNA interaction algorithms, e.g. [10].

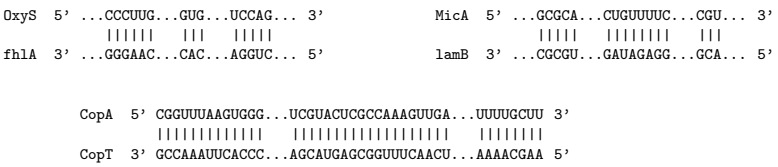


**Fig. 6.** Known pairs of interacting RNAs

**Structural Separation.** The yeast snRNA complex U2-U6 is necessary for the splicing of a specific mRNA intron [23]. Only the preserved regions of the intron are considered, which consist of two structurally autonomous parts, resulting in an instance with a total of four RNAs, U2, U6, I1, and I2. Six RNAs are used: CopA, CopT, and the four mentioned RNAs. The interest here is to see whether the algorithm can separate the CopA-CopT complex from that of yeast. The algorithm is performed with $k = 3$ and gap filling. The solution with the largest weight successfully predicts and separates the RNA complex CopA-CopT of Figure 6 from the RNA structure shown in Figure 7a for the U2-U6 complex in the splicing of its intron.

### 4.2 Multiple Sub-optimal Solutions

We now use the algorithm of Section 3.3 with an appropriate threshold $T$ to generate enough solutions. Additional parameters for this algorithms are: the distance function $d(S, C)$, the threshold $D$ for adding a solution to a cluster, and whether the cluster representative is the best solution in the cluster or the consensus of the cluster (refer to Section 3.3.3 for computing distances). The choice of these parameters will be given for each scenario. Only the best solution in each cluster is reported (see Figure 5 for detail).

**Helices and Co-axial Stacking.** The U2-U6 complex in yeast has been reported to have two distinct experimental structures, e.g. [24]. In one conformation, U2 and U6 interact to form helix Ia (interaction as in Figure 7c). In another conformation, the interaction reveals a structure containing an additional helix, helix Ib. It has been conjectured in [25] that co-axial stacking is essential for the stabilization of helix Ia in U2-U6 and, therefore, inhibition of the co-axial stacking, possibly by protein binding, may activate the second conformation. Regardless of what underlying mechanisms are responsible for this conformational switch, our sub-optimal solutions cluster in a way that reveal the two conformations (Figure 7).

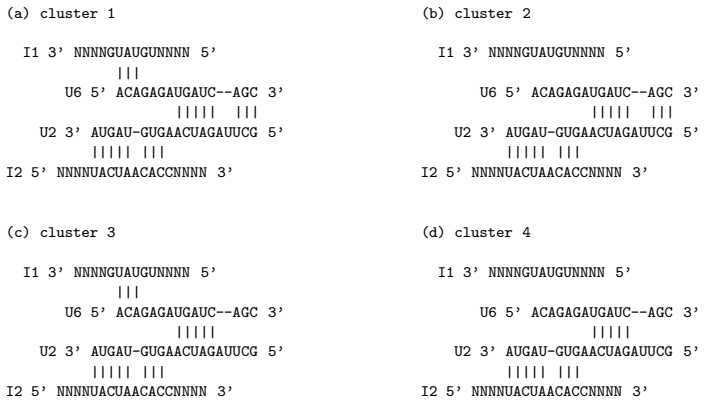


**Fig. 7.** U2 and U6 truncated up to helix Ib. Algorithm performed with the following parameters: distance is Jaccard, threshold=0.15, representative is consensus. (a) Helices Ia and Ib with correct binding of introns. (b) same as (a) with I1 not binding. (c) Helix Ia only with correct binding of introns. (d) Same as (c) with I1 not binding.

**Artifact Interactions and Reversible Kissing Loops.** Due to the optimization nature of the problem, it is sometimes easy to pick up interactions that are biologically unreal. This is because dropping these interactions from the solution would make it less optimal. The third interaction window of CopA-CopT in Figure 6 is an example of such an artifact. As shown in Figure 8 on the Left, our second cluster of sub-optimal solutions succeeds in dropping this window.

Reversible kissing loops represent an even harder mechanism to capture by optimization. With this mechanism, the initial kissing complex occurs between a subset of loop bases in both RNAs, but this interaction is fully reversible and very unstable [26]. Therefore, in the final interaction, the kissing loop will be missing few bases towards its center. An example of this scenario is the middle interaction window of CopA-CopT in Figure 6 and Figure 8 on the Left (considering the folding pattern of CopA and CopT reveals that this interaction window is a kissing loop). By isolating this window and generating sub-optimal solutions, our third cluster starts to reveal a separation of the interaction close to the center, as shown in Figure 8 on the Right.

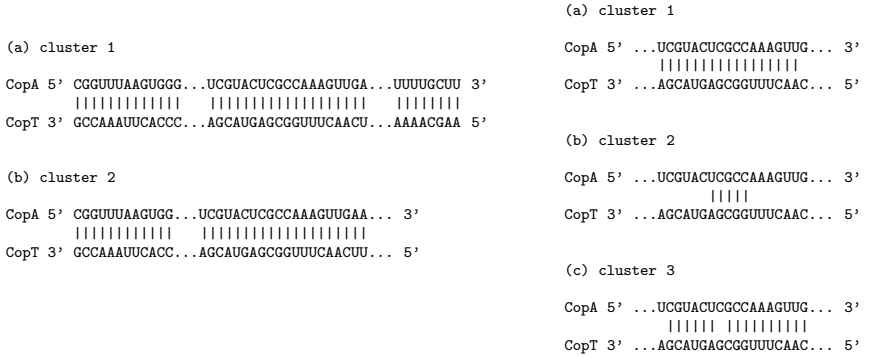


**Fig. 8.** Left: Algorithm performed with the following parameters: distance is average of Jaccard and Levenshtein, threshold=0.25, representative is best. (a) As in Figure 6. (b) A less optimal but realistic structure in which the third interaction window is dropped. Right: CopA-CopT complex truncated to its middle window. Algorithm performed with the following parameters: distance is average of Jaccard and Levenshtein, threshold=0.3, representative is best. The third cluster starts to reveal a separation (reversible kissing loop) in the middle interaction window.

## 5 Conclusion

While RNA-RNA interaction algorithms exist, they are not suitable for predicting RNA structures with more than two RNAs; for instance, treating the RNAs pairwise may not lead to the best global structure. Moreover, the best structure may not be the real structure, and the real structure may not be unique. In this work, we build on our recent formulation for multiple RNA interaction as a combinatorial optimization problem, and extend it to produce multiple sub-optimal solutions. Our experiments reveal that such an approach can provide several candidate structures when they exist, e.g. the U2-U6 complex in the spliceosome of yeast, and find realistic structures that are not necessarily optimal in the computational sense, e.g. CopA-CopT in E. Coli.

## References

1. Pervouchine, D.D.: Iris: Intermolecular RNA interaction search. In: 15th International Conference on Genome Informatics (2004)
2. Alkan, C., Karakoc, E., Nadeau, J.H., Sahinalp, S.C., Zhang, K.: RNA-RNA interaction prediction and antisense RNA target search. Journal of Computational Biology 13(2) (2006)
3. Mneimneh, S.: On the approximation of optimal structures for RNA-RNA interaction. IEEE/ACM Transactions on Computational Biology and Bioinformatics (2009)

4. Meyer, I.M.: Predicting novel RNA-RNA interactions. Current Opinions in Structural Biology 18 (2008)
5. Kolb, F.A., Malmgren, C., Westhof, E., Ehresmann, C., Ehresmann, B., Wagner, E.G.H., Romby, P.: An unusual structure formed by antisense-target RNA binding involves an extended kissing complex with a four-way junction and a side-by-side helical alignment. RNA Society (2000)
6. Argaman, L., Altuvia, S.: fhla repression by oxys: Kissing complex formation at two sites results in a stable antisense-target RNA complex. Journal of Molecular Biology 300 (2000)
7. Muckstein, U., Tafer, H., Hackermuller, J., Bernhart, S.H., Stadler, P.F., Hofacker, I.L.: Thermodynamics of RNA-RNA binding. In: Journal of Bioinformatics (2006)
8. Chitsaz, H., Backofen, R., Sahinalp, S.C.: biRNA: Fast RNA-RNA binding sites prediction. In: Salzberg, S.L., Warnow, T. (eds.) WABI 2009. LNCS, vol. 5724, pp. 25–36. Springer, Heidelberg (2009)
9. Chitsaz, H., Salari, R., Sahinalp, S.C., Backofen, R.: A partition function algorithm for interacting nucleic acid strands. Journal of Bioinformatics (2009)
10. Salari, R., Backofen, R., Sahinalp, S.C.: Fast prediction of RNA-RNA interaction. Algorithms for Molecular Biology 5(5) (2010)
11. Huang, F.W.D., Qin, J., Reidys, C.M., Stadler, P.F.: Partition function and base pairing probabilities for RNA-RNA interaction prediction. Journal of Bioinformatics 25(20) (2009)
12. Li, A.X., Marz, M., Qin, J., Reidys, C.M.: RNA-RNA interaction prediction based on multiple sequence alignments. In: Journal of Bioinformatics (2010)
13. Sun, J.S., Manley, J.L.: A novel u2-u6 snrna structure is necessary for mammalian MRNA splicing. Genes and Development 9 (1995)
14. Andronescu, M., Chuan, Z.Z., Codon, A.: Secondary structure prediction of interacting RNA molecules. Journal of Molecular Biology 345(5) (2005)
15. Dirks, R.M., Bois, J.S., Schaffer, J.M., Winfree, E., Pierce, N.A.: Thermodynamic analysis of interacting nucleic acid strands. SIAM Review 49(1) (2007)
16. McCaskill, J.S.: The equilibrium partition function and base pair binding probabilities for RNA secondary structure. Journal of Biopolymers 29(6-7) (1990)
17. Chen, H.L., Codon, A., Jabbari, H.: An $o(n^5)$ algorithm for mfa prediction of kissing hairpins and 4-chains in nucleic acids. Journal of Computational Biology 16(6) (2009)
18. Ahmed, S.A., Mneimneh, S., Greenbaum, N.L.: A combinatorial approach for multiple RNA interaction: Formulations, approximations, and heuristics. In: Du, D.-Z., Zhang, G. (eds.) COCOON 2013. LNCS, vol. 7936, pp. 421–433. Springer, Heidelberg (2013)
19. Cormen, T., Leiserson, C.E., Rivest, R.L., Stein, C.: Approximation Algorithms in Introduction to Algorithms. MIT Press (2010)
20. Tong, W., Goebel, R., Liu, T., Lin, G.: Approximation algorithms for the maximum multiple RNA interaction problem. In: Widmayer, P., Xu, Y., Zhu, B. (eds.) COCOA 2013. LNCS, vol. 8287, pp. 49–59. Springer, Heidelberg (2013)
21. Jaccard, P.: Bulletin de la Societe Vaudoise des Sciences Naturelles 38(69) (1902)
22. Levenshtein, V.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady (10), 70710 (1966)
23. Newby, M.I., Greenbaum, N.L.: A conserved pseudouridine modification in eukaryotic u2 snrna induces a change in branch-site architecture. RNA 7(6), 833–845 (2001)

24. Sashital, D.G., Cornilescu, G., Butcher, S.E.: U2-u6 RNA folding reveals a group ii intron-like domain and a four-helix junction. Nature Structural and Molecular Biology 11(12) (2004)
25. Cao, S., Chen, S.J.: Free energy landscapes of RNA/RNA complexes. Journal of Molecular Biology (357), 292–312 (2006)
26. Kolb, F.A., Slagter-Jager, J.G., Ehresmann, B., Ehressmann, C., Westhof, E., Gerhart, E., Wagner, H., Romby, P.: Progression of a loop-loop complex to a four-way junction is crucial for the activity of a regulatory antisense RNA. The EMBO Journal 19(21), 5905–5915 (2000)