Approximating the 2-Interval Pattern problem

Maxime Crochemore*, Danny Hermelin**,
Gad M. Landau***, Dror Rawitz†, and Stéphane Vialette‡

Abstract. We address the problem of approximating the 2-Interval Pattern problem over its various models and restrictions. This problem, motivated by RNA secondary structure prediction, asks to find a maximum cardinality subset of a 2-interval set with respect to some prespecified geometric constraints. We present several constant factor approximation algorithms whose performance guarantee depends on the different possible restrictions imposed on the input 2-interval set. In addition, we show that our results extend to the weighted variant of the problem.

Key Words: 2-interval, RNA secondary structure prediction, combinatorial approximation algorithms.

1 Introduction

The Ribonucleic acid (RNA) is a family of molecules which have several important functions in the cell. An RNA molecule is a single stranded molecule which can be viewed as a linear sequence consisting of four nucleotides: Adenine (A), Cytosine (C), Guanine (G), and Uracil (U). The pairs of nucleotides A-U and C-G are known as complementary nucleotide pairs which often link together by their phosphodiester bonds to form a three dimensional folding structure. This folding structure is captured in many ways, in what is called the *secondary structure*, the set of all hydrogen bonds formed by the nucleotides of the molecule. It is widely believed that for many interesting families of RNA molecules, the functionality of the molecule depends mostly on its secondary structure [18]. Since current biological methods for extracting sequential data exceed by far methods for extracting structural data, there is a need to predict the secondary structure of an RNA given its sequence of nucleotides. This is known as *secondary structure prediction* [21].

RNA secondary structure prediction usually focuses on predicting the structure with minimum free energy [21], *i.e.* the stablest structure possible, where each nucleotide is assumed to bond with at most one other nucleotide. There are many approaches to determine the free energy of a given structure. One simplified approach, chosen also in [15], is to consider only the helices of the structure, as they are believed to contribute to the stability of the structure in the most significant way. A *helix* in an RNA molecule consists of two disjoint consecutive sequences of nucleotides, where almost every nucleotide in one sequence is paired with another nucleotide in the second sequence.

In [20], a geometric representation of a helix in an RNA molecule is proposed by means of a natural generalization of an interval, namely a 2-interval. There, intervals and 2-intervals represent, respectively, sequences of contiguous nucleotides and possible pairings between such sequences in the RNA molecule (see Figure 1). The prediction of a secondary structure under this approach consists of two stages. In the first stage, the sequence of molecules is scanned in order to build a

* Institut Gaspard-Monge, Université de Marne-la-Vallée, France, and Department of Computer Science, King's Collage, London, UK. Partially supported by CNRS, France, and the French Ministry of Research through ACI NIM. **maxime.crochemore@univ-mlv.fr**.

** Department of Computer Science, University of Haifa, Israel. **danny@cri.haifa.ac.il**.

*** Department of Computer Science, University of Haifa, Israel, and Department of Computer and Information Science, Polytechnic University, NY, USA. Partially supported by the Israel Science Foundation grant 282/01. **landau@cs.haifa.ac.il**.

† Caesarea Rothschild Institute, University of Haifa, Israel. **rawitz@cri.haifa.ac.il**.

‡ Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud, France. **vialette@lri.fr**.

set of 2-intervals which correspond to all helixes that could be involved in the molecule's secondary structure. In the second stage, a pairwise disjoint subset of 2-intervals is sought for, possibly under some additional constraints, so as to serve as an estimate of the actual secondary structure of the molecule. The problem we study in this paper, *i.e.* 2-Interval Pattern, is concerned with the second stage of this process.

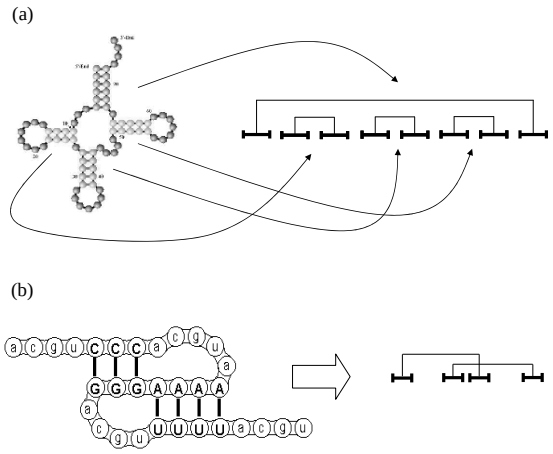

Fig. 1. Two segments of RNA molecules and the set of corresponding 2-intervals. In (a), the secondary structure is pseudoknot-free. In (b), any pair of bonds $C-G$ and $A-U$ is a pseudoknot.

A 2-*interval* [14, 19] is the union of two disjoint intervals defined over a single line. Throughout the paper, a 2-interval is denoted by $D = (I, J)$ where I and J are two (closed) intervals defined over a single line such that I is completely to the left of J. Two 2-intervals $D_1 = (I_1, J_1)$ and $D_2 = (I_2, J_2)$ are *disjoint*, if both 2-intervals share no common point, that is, if $(I_1 \cup J_1) \cap (I_2 \cup J_2) = \emptyset$. For such disjoint pairs of 2-intervals, three natural binary relations are of special interest.

Definition 1 (Relations between 2-intervals). *Let $D_1 = (I_1, J_1)$ and $D_2 = (I_2, J_2)$ be two disjoint 2-intervals. Then*

- $D_1 < D_2$ *(D_1* precedes *D_2), if $I_1 < J_1 < I_2 < J_2$.*
- $D_1 \sqsubset D_2$ *(D_1 is* nested *in D_2), if $I_2 < I_1 < J_1 < J_2$.*
- $D_1 \between D_2$ *(D_1* crosses *D_2), if $I_1 < I_2 < J_1 < J_2$.*

A pair of 2-intervals D_1 and D_2 is *R-comparable* for some $R \in \{<, \sqsubset, \between\}$, if either $(D_1, D_2) \in R$ or $(D_2, D_1) \in R$. A set of 2-intervals $\mathcal{D}$ is $\mathcal{R}$-comparable for some $\mathcal{R} \subseteq \{<, \sqsubset, \between\}$, $\mathcal{R} \neq \emptyset$, if any pair of distinct 2-intervals in $\mathcal{D}$ is R-comparable for some $R \in \mathcal{R}$. The non-empty subset $\mathcal{R}$ is called a *model*. Note that any two disjoint 2-intervals are R-comparable for some $R \in \{<, \sqsubset, \between\}$. Equivalently, any pairwise disjoint subset of $\mathcal{D}$ is $\{<, \sqsubset, \between\}$-comparable.

Definition 2 (The 2-Interval Pattern problem [6, 20]). *Let $\mathcal{D}$ be a set of 2-intervals and let $\mathcal{R} \subseteq \{<, \sqsubset, \between\}$, $\mathcal{R} \neq \emptyset$, be a given model. The* 2-Interval Pattern *problem asks to find a maximum cardinality $\mathcal{R}$-comparable subset of $\mathcal{D}$.*

By the above definition, any solution for the 2-Interval Pattern problem over a model $\mathcal{R}$ corresponds to a secondary structure constrained by $\mathcal{R}$. For example, a solution for the 2-Interval Pattern problem over the $\{<, \sqsubset\}$ model corresponds to a pseudoknot-free structure. A *pseudoknot*

in an RNA sequence $S = s_1, s_2, \ldots, s_n$ is composed of two interleaving nucleotide pairings (s_i, s_j) and $(s_{i'}, s_{j'})$ such that $i < i' < j < j'$ (see Figure 1).

Definition 3 (Restrictions for 2-interval sets). *Let $\mathcal{D}$ be a set of 2-intervals and let $\mathcal{S}(\mathcal{D})$ be the set of intervals involved in $\mathcal{D}$.*

- *$\mathcal{D}$ is a* point 2-interval set *if all intervals in $\mathcal{S}(\mathcal{D})$ are pairwise disjoint (note that in this case, all intervals in $\mathcal{S}(\mathcal{D})$ may be considered as points).*
- *$\mathcal{D}$ is a* unitary 2-interval set *if all intervals in $\mathcal{S}(\mathcal{D})$ are of equal length.*
- *$\mathcal{D}$ is a* balanced 2-interval set *if any 2-interval in $\mathcal{D}$ is a pair of two intervals of equal length.*
- *$\mathcal{D}$ is an* unlimited 2-interval set *if none of the above restrictions are imposed.*

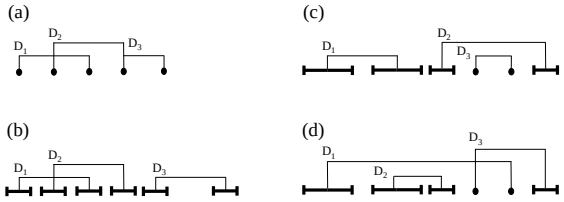

Fig. 2. The different possible restrictions considered for 2-interval sets. Intervals are represented by dark thick lines and points, and 2-intervals are represented by a thin line connecting two intervals. (a) A point 2-interval set where $D_1 \between D_2$ and $D_1 < D_3$. The pair of 2-intervals D_2 and D_3 are not disjoint and thus are not comparable by any relation. (b) A unitary 2-interval set where $D_1 \between D_2$, $D_1 < D_3$, and $D_2 < D_3$. (c) A balanced 2-interval set where $D_3 \sqsubset D_2$. The entire set is $\{<, \sqsubset\}$-comparable. (d) An unlimited $\{<, \sqsubset, \between\}$-comparable 2-interval set.

The left part of Table 1 depicts the current state of the art for the 2-Interval Pattern problem in terms of exact algorithms. In [20], 2-Interval Pattern over $\{\sqsubset, \between\}$ and $\{<, \sqsubset, \between\}$ is proved to be **NP**-hard even for unitary 2-interval sets. The proof for the $\{<, \sqsubset, \between\}$ model is obtained as a direct consequence of the **APX**-hardness result for the Maximum Independent Set problem in t-interval graphs [5]. The results in [5] also provide approximation algorithms for this model. In [6], an **NP**-hardness result for the $\{<, \between\}$ model restricted to unitary 2-interval sets is given. The time complexity for this same model when the input is restricted to point 2-interval sets is still unknown. These results imply that in practical terms, secondary structures containing pseudoknots are hard to predict in our suggested mathematical model. This is consistent with previously known **NP**-hardness results for RNA secondary structures prediction in other models considering arbitrary pseudoknots [1, 15, 16]. Other works which are similar to our line of research include the Arc-Preserving Subsequence (APS) and Longest Arc-Preserving Common Subsequence (LAPCS) problems studied in [8, 13], and the Contact Map Overlap problem described in [11].

1.1 Our results

In this paper we focus on the three **NP**-hard models of the 2-Interval Pattern problem. More specifically, we design constant factor approximation algorithms for the $\{<, \sqsubset, \between\}$, $\{\sqsubset, \between\}$, and $\{<, \between\}$ models. The approximation factors obtained by our algorithms vary depending on the given model and the restriction imposed on the input set of 2-intervals. Furthermore, we complement the **APX**-hardness result for the $\{<, \sqsubset, \between\}$ model [5, 20], with an **APX**-hardness result for the $\{\sqsubset, \between\}$ model.

2-Interval Pattern - Classical Complexity				
Model	Unlimited	Balanced	Unitary	Point
$\{<, \sqsubset, \between\}$	**NP**-complete [5, 20]			$\mathcal{O}(n\sqrt{n})$ [20]
$\{\sqsubset, \between\}$	**NP**-complete [20]			$\mathcal{O}(n^2\sqrt{n})$ [6]
$\{<, \between\}$	**NP**-complete [6]			?
$\{<, \sqsubset\}$	$\mathcal{O}(n^2)$ [20]			
$\{\between\}$	$\mathcal{O}(n^2 \log n)$ [20]			
$\{\sqsubset\}$	$\mathcal{O}(n \log n)$ [6]			
$\{<\}$	$\mathcal{O}(n \log n)$ [20]			

2-Interval Pattern- Approximation Factors				
Model	Unlimited	Balanced	Unitary	Point
$\{<, \sqsubset, \between\}$ (Section 2)	4^a [5]	4^b	3^b [5]	–
$\{\sqsubset, \between\}$ (Section 3)	4^a	4^c	3^c	–
$\{<, \between\}$ (Section 4)	6^b	4^b	3^b	3^b

[a] Polynomial-time algorithm (linear programming).
[b] $\mathcal{O}(n \lg n)$ time algorithm.
[c] $\mathcal{O}(n^2 \lg n)$ time algorithm.

Table 1. The 2-Interval Pattern problem over it's various models and restrictions. Left part: complexity results for the 2-Interval Pattern problem, where $n = |\mathcal{D}|$. The 2-Interval Pattern problem for the $\{<, \between\}$ model restricted to point 2-interval sets is not known to be in **P** or **NP**-complete. Right part: The approximation factors we obtain for the 2-Interval Pattern problem supporting the idea that the problem has varying approximation quality depending on the different possible restrictions imposed on the input 2-interval set.

Another contribution of this paper is a new restriction on the input set of 2-intervals, namely the balanced restriction. By definition, unitary 2-interval sets are also balanced but the converse is not necessarily true. Consequently, the above mentioned hardness results also hold for the balanced case, and moreover, balanced 2-interval sets introduce a new combinatorial object which requires particular consideration. Our motivation for considering balanced 2-interval sets is very natural in the biological setting of the 2-Interval Pattern problem. Indeed in our suggested mathematical model, a 2-interval corresponds to a helix in a RNA secondary structure, which is often considered to be composed of two disjoint sequences of nucleotides of equal length.

Finally, we introduce a weighted variant of the 2-Interval Pattern problem, in which each 2-interval is associated with a weight, and the goal is to find a maximum weight subset of a 2-interval set with respect to a prespecified model. Here, one can for instance, weight a 2-interval by the total sum of the lengths of its intervals, thereby allowing more refined solutions in the biological application of the problem. We show that our results can be extended to the weighted variant, while still maintaining the same approximation factors.

This paper is organized as follows. In Section 2, we consider the 2-Interval Pattern problem over the the $\{<, \sqsubset, \between\}$ model. In Section 3, we describe an approximation algorithm for the problem over the $\{\sqsubset, \between\}$ model. In Section 4, the $\{<, \between\}$ model is considered, and different approximation algorithms are introduced for all possible restrictions imposed on the input. In Section 5 we show that our results extend to the Weighted 2-Interval Pattern problem.

2 Approximation algorithms for the $\{<, \sqsubset, \between\}$ model.

We begin by considering the 2-Interval Pattern problem over the $\{<, \sqsubset, \between\}$ model. Recall that in this case, given an input set of 2-intervals $\mathcal{D}$, the problem asks to find a maximum $\{<, \sqsubset, \between\}$-comparable subset of $\mathcal{D}$, or equivalently, to find a maximum pairwise disjoint subset of $\mathcal{D}$.

For point 2-intervals sets, 2-Interval Pattern can be solved in polynomial time by maximum matching [20]. For unitary 2-interval sets, the problem is already **APX**-hard [5]. Furthermore, the results in [5] also yield approximation algorithms for our case, directly implying the following.

Proposition 1 ([5]). *The* 2-Interval Pattern *problem over the* $\{<, \sqsubset, \between\}$ *model can be approximated within a factor of 4 when restricted to unlimited* 2*-interval sets, and a factor of 3 when restricted to unitary interval sets.*

The approximation algorithm given in [5] that solves the case of unitary 2-interval sets can be executed in $\mathcal{O}(n \lg n)$ time, where n is the size of the input set of 2-intervals. However, the algorithm for unlimited 2-interval sets uses linear programming techniques, which in practice are very often too time costly. Clearly, the case of balanced 2-interval sets lies between the two cases and is arguably the most biologically important case. In the rest of this section, we describe an $\mathcal{O}(n \lg n)$ time 4-approximation algorithm for balanced 2-intervals sets.

Given any balanced 2-interval set $\mathcal{D}$, the *smallest* 2-interval in $\mathcal{D}$ is the 2-interval with the shortest left (or right, as they are both of equal length) interval among all left intervals involved in $\mathcal{D}$ (ties are broken arbitrarily). We suggest a simple greedy algorithm that repeatedly picks the smallest 2-interval in the input, adds it to the solution, and omits all other 2-intervals in the input which intersect it. A schematic description of this algorithm, which we call Bal-$\{<, \sqsubset, \between\}$-Approx, is given in Figure 3.

Algorithm Bal-$\{<, \sqsubset, \between\}$-Approx($\mathcal{D}$)

Data : A set of balanced 2-intervals $\mathcal{D}$.
Result : A $\{<, \sqsubset, \between\}$-comparable subset of $\mathcal{D}$.
begin
  while $\mathcal{D} \neq \emptyset$ **do**
    1. Let D_0 be the smallest 2-interval in $\mathcal{D}$.
    2. Add D_0 to the solution.
    3. Omit D_0 and all 2-intervals intersecting D_0 from $\mathcal{D}$.
  end
  return *the 2-intervals chosen for the solution.*
end

Fig. 3. A schematic description of algorithm Bal-$\{<, \sqsubset, \between\}$-Approx.

Lemma 1. *Algorithm Bal-$\{<, \sqsubset, \between\}$-Approx achieves an approximation factor of 4 for the* 2-Interval Pattern *problem over the general model, restricted to balanced 2-interval sets.*

Proof. First note that Bal-$\{<, \sqsubset, \between\}$-Approx computes a $\{<, \sqsubset, \between\}$-comparable set of 2-intervals by construction. Now, let $\mathcal{D}$ be the set of remaining 2-intervals at any arbitrary iteration of the algorithm, and let $D_0 \in \mathcal{D}$ be the smallest 2-interval at this iteration. Since D_0 is the smallest 2-interval in $\mathcal{D}$, no interval involved in $\mathcal{D}$ can be properly contained in the left or right interval of D_0. Thus, amongst all the 2-intervals omitted at this iteration, there can be no more than four 2-intervals which are mutually pairwise disjoint. It follows that at most four 2-intervals from any optimal solution are omitted at this iteration. Applying this argument for all iterations of the algorithm yields the desired approximation factor guarantee. □

Implementation remark. Note that as stated above, algorithm Bal-$\{<, \sqsubset, \between\}$-Approx runs in $\mathcal{O}(n^2)$ time. In the following we show that omitting 2-intervals which are not in the solution in a slightly different way, allows reducing this time bound to $\mathcal{O}(n \lg n)$.

First, we sort $\mathcal{D}$ from the smallest 2-interval to the largest one (*i.e.* the 2-interval with the largest left or right interval). Furthermore, we use an auxiliary binary search tree that maintains all endpoints of 2-intervals in our solution. The main idea is that in step 3 of each iteration, we omit only D_0. Any 2-interval intersecting D_0 is omitted at a later stage. In step 1 of each iteration, we first check if the current D_0 is one of those 2-intervals that should have been omitted earlier, and it is omitted in such a case. Otherwise, in step 2 we add D_0 to the solution, and we also insert its four endpoints to the auxiliary search tree.

The only non-trivial computation is the one in step 1 that checks if D_0 should have been omitted earlier. Since all 2-intervals in the solution are smaller than the current D_0, if D_0 has to be omitted, then at least one of its intervals contains an endpoint of one of the 2-intervals in the solution. This can be checked using two $\mathcal{O}(\lg n)$ query operations in our search tree.

Time complexity. When implemented as above, algorithm Bal-$\{<, \sqsubset, \between\}$-Approx runs in $\mathcal{O}(n \lg n)$ time. Indeed, sorting the 2-intervals requires $\mathcal{O}(n \lg n)$ time. Furthermore, each iteration can be done in $\mathcal{O}(\lg n)$ time, since we perform a constant number of insertion and query operations on our search, and all other operations require $\mathcal{O}(1)$ time.

3 An approximation algorithm for the $\{\sqsubset, \between\}$ model.

We next consider the 2-Interval Pattern problem over the $\{\sqsubset, \between\}$ model. Recall that for point 2-interval sets there exists an $\mathcal{O}(n^2\sqrt{n})$ algorithm for the problem, while for unitary 2-intervals, the problem is already **NP**-complete [20]. We begin our discussion in this section, by introducing a single constant approximation algorithm, which achieves different approximation factors, depending on the different possible restrictions imposed on the input 2-interval set. Following this, we show that 2-Interval Pattern over $\{\sqsubset, \between\}$ is in fact **APX**-hard, even in the case where the input is restricted to a unitary 2-interval set.

Our algorithm is a generalization of the $\mathcal{O}(n^2\sqrt{n})$ algorithm devised in [6] for 2-Interval Pattern over $\{\sqsubset, \between\}$ restricted to point 2-interval sets. As in [6], the notion of *interval graphs* is used extensively throughout the section. An interval graph is an intersection graph of a finite family of intervals, all defined over a single line [12, 17].

Given a 2-interval $D = (I, J)$, let $C(D)$ denote the smallest interval that covers D, *i.e.*, $C(D) = [l(I) : r(J)]$ where $l(I)$ and $r(J)$ are the left and right endpoints of I and J, respectively. Blin *et al.* [6] called $C(D)$ the *covering interval* of D. They also observed that any pair of disjoint 2-intervals are $\{\sqsubset, \between\}$-comparable if and only if their corresponding covering intervals intersect. Thus, given a set of 2-intervals $\mathcal{D}$, and the set $\mathcal{C}(\mathcal{D})$ of all covering intervals of 2-intervals in $\mathcal{D}$, any $\{\sqsubset, \between\}$-comparable subset $\mathcal{D}' \subseteq \mathcal{D}$ corresponds to a pairwise intersecting subset of $\mathcal{C}' \subseteq \mathcal{C}(\mathcal{D})$. However, the converse is not true as a pair of non-disjoint 2-intervals have corresponding intersecting covering intervals as well. Hence, a pairwise intersecting subset of $\mathcal{C}(\mathcal{D})$ can contain corresponding 2-intervals which are non-disjoint in $\mathcal{D}$. Figure 4 depicts this relationship between 2-intervals and their corresponding covering intervals.

Let $\mathcal{D}$ be the input set of 2-intervals and $\mathcal{C}(\mathcal{D})$ be the set of covering intervals of all 2-intervals in $\mathcal{D}$. First, we construct the interval graph $\Omega_{\mathcal{C}(\mathcal{D})}$ of $\mathcal{C}(\mathcal{D})$. Any pair of 2-intervals with covering intervals in a clique of $\Omega_{\mathcal{C}(\mathcal{D})}$, are either nesting or crossing (but not preceding), or they are non-disjoint. Now, let OPT denote a maximum cardinality $\{\sqsubset, \between\}$-comparable subset of $\mathcal{D}$, and let $\mathcal{C}(OPT)$ be the set of covering intervals of OPT. The subgraph of $\Omega_{\mathcal{C}(\mathcal{D})}$ which corresponds to $\mathcal{C}(OPT)$ is a clique, and is thus a subset of some maximal (in inclusion order) clique of $\Omega_{\mathcal{C}(\mathcal{D})}$. Furthermore, any 2-interval with a covering interval in this clique and not in OPT is necessarily non-disjoint with at least one of the 2-intervals in OPT.

Observation 1. *OPT is a maximum pairwise disjoint subset of a set of 2-intervals $\mathcal{D}'$ ($OPT \subseteq \mathcal{D}' \subseteq \mathcal{D}$), such that $\mathcal{C}(\mathcal{D}')$, the set of covering intervals of $\mathcal{D}'$, corresponds to a maximal clique in $\Omega_{\mathcal{C}(\mathcal{D})}$.*

Since $\Omega_{\mathcal{C}(\mathcal{D})}$ is an interval graph, it has at most $|V(\Omega_{\mathcal{C}(\mathcal{D})})| = |\mathcal{D}|$ maximal cliques, and these can be computed in polynomial time [10]. Furthermore, given the 2-intervals which corresponds to

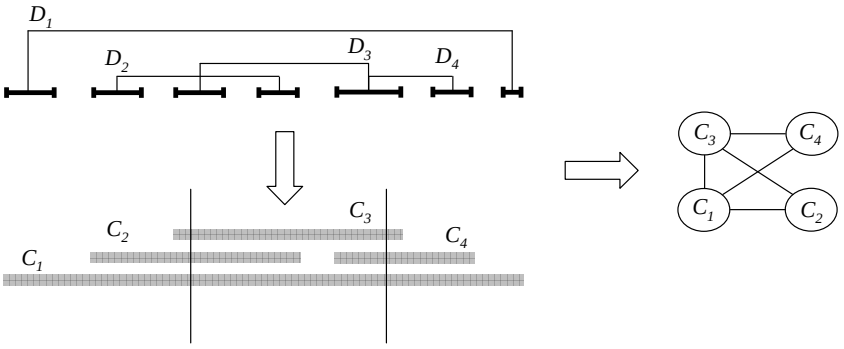

Fig. 4. A set of 2-intervals, their corresponding covering intervals, and the interval graph of these covering intervals. Subsets $\{C_1, C_2, C_3\}$ and $\{C_1, C_3, C_4\}$ are the maximal intersecting subsets of $\mathcal{C}(\mathcal{D})$ and therefore are maximal cliques in the interval graph. Subset $\{D_1, D_2, D_3\}$ is $\{\sqsubset, \emptyset\}$-comparable in $\mathcal{D}$, while subset $\{D_1, D_3, D_4\}$ is not because D_3 and D_4 intersect.

a maximal clique in $\Omega_{\mathcal{C}(\mathcal{D})}$, one can use the algorithms in Section 2 to find an approximation of the maximum pairwise disjoint subset of these 2-intervals. A detailed schematic description of our algorithm, which is called $\{\sqsubset, \emptyset\}$-Approx, is given in Figure 5.

Algorithm $\{\sqsubset, \emptyset\}$-Approx($\mathcal{D}$)

Data : A set of 2-intervals $\mathcal{D}$.
Result : A $\{\sqsubset, \emptyset\}$-comparable subset of $\mathcal{D}$.
begin
  1. Construct $\mathcal{C}(\mathcal{D})$, the set of covering intervals of all 2-intervals in $\mathcal{D}$.
  2. Construct $\Omega_{\mathcal{C}(\mathcal{D})}$, the interval graph of $\mathcal{C}(\mathcal{D})$.
  3. Compute all maximal cliques of $\Omega_{\mathcal{C}(\mathcal{D})}$ using [10].
  4. foreach *maximal clique* C *of* $\Omega_{\mathcal{C}(\mathcal{D})}$ **do**
    (a) Compute $\mathcal{D}_C \subseteq \mathcal{D}$, the 2-intervals with covering intervals in C.
    (b) Approximate the maximum pairwise disjoint subset of $\mathcal{D}_C$ using the algorithms described in the previous section.
  end
  return *the largest pairwise disjoint subset found in step 4(b).*
end

Fig. 5. A schematic description of algorithm $\{\sqsubset, \emptyset\}$-Approx.

Lemma 2. *Algorithm* $\{\sqsubset, \emptyset\}$*-Approx is a* 4*-approximation* (3*-approximation*) *algorithm for the* 2-Interval Pattern *problem for unlimited and balanced* (*unitary*) 2*-interval sets.*

Proof. Immediate from the above discussion and from Proposition 1 and Lemma 1. □

Time complexity. The number of sub-procedure invocations in step 4(b) of $\{\sqsubset, \emptyset\}$-Approx is bounded by $\mathcal{O}(n)$ where n denotes the size of the input set. Also, generating all maximal cliques of $\Omega_{\mathcal{C}(\mathcal{D})}$ can be done in $\mathcal{O}(n^2)$ time. Hence, we have a super-quadratic running time of $\mathcal{O}(n^2 \lg n)$ for unitary and balanced 2-interval sets, and a polynomial running time for unlimited 2-interval sets [5].

Next we show that 2-Interval Pattern over $\{\sqsubset, \emptyset\}$ is **APX**-hard. For this, we consider a special class of intersection graphs, called *2-union graphs* [5]. A 2-union graph is the union of two

interval graphs with the same vertex set. Thus, given two distinct lines, a 2-union graph is an intersection graph of a family of pairs of intervals, where each pair consists of two intervals, one on each line. Two vertices are connected in the graph if, and only if, the intervals of the pairs are intersecting on at least one of these lines.

In [5], Bar-Yehuda *et al.* proved that the Maximum Independent Set problem for 2-union graphs is **APX**-hard, even if the input includes a unitary representation of the graph. That is, it includes a family of pairs of intervals, such that each interval in the family is of equal length. We show that finding a maximum pairwise disjoint subset in such a family, and hence a maximum independent set in the graph, reduces to finding a maximum $\{\sqsubset, \between\}$-comparable subset in a set of unitary 2-intervals.

Let G be a 2-union graph and let $R(G)$ be its unitary representation. Construct a set of 2-intervals $\mathcal{D}$ by considering the two lines over which the intervals in $R(G)$ are defined over, as two disjoint segments of the same line (see Figure 6). Clearly G is also the intersection graph of $\mathcal{D}$. Furthermore, $\mathcal{D}$ does not contain any pair of 2-intervals which is $\{<\}$-comparable. Hence, any independent set in G corresponds to a $\{\sqsubset, \between\}$-comparable subset of $\mathcal{D}$ of equal size.

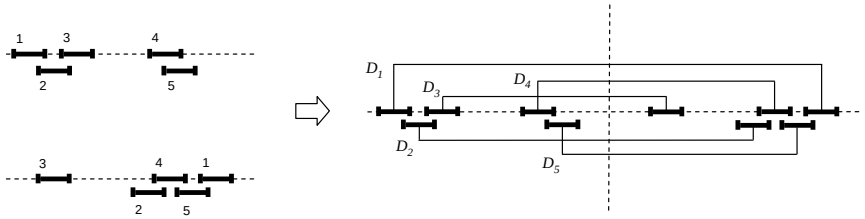

Fig. 6. A unitary representation of a 2-union graph transformed into a unitary 2-interval set.

Corollary 1. *The* 2-Interval Pattern *problem over the* $\{\sqsubset, \between\}$ *model is* **APX**-*hard, even when restricted to unitary 2-interval sets.*

4 Approximation algorithms for the $\{<, \between\}$ model.

We now turn to considering the 2-Interval Pattern problem over the $\{<, \between\}$ model. Recall that the problem is known to be **NP**-hard for unitary 2-interval sets, while for point 2-interval sets the problem is not known to be polynomial-time solvable [6]. Thus, in this section we consider all possible restrictions for this model. More specifically, we design a 3-approximation algorithm for unitary 2-interval sets which is also a 2-approximation algorithm for point 2-interval sets. We later slightly modify this algorithm to obtain a 5-approximation algorithm for balanced 2-interval sets. Finally, we introduce a slightly more involved modification which yields a 6-approximation algorithm for the unlimited case. Determining whether or not the problem is **APX**-hard, and if so under what restrictions, is left as an open problem.

Throughout the section, we will use the notion of *trapezoid graph* [7, 9]. Consider two intervals, I' and J', defined over two distinct horizontal lines. The trapezoid $T = (I', J')$ is the convex set of points bounded by I' and J', and the two line segments connecting the right and left endpoints of I' and J'. We call I' the *bottom interval* and J' the *top interval* of T. A family of trapezoids is a finite set of trapezoids which are all defined over the same two horizontal lines. The above definitions imply, that two distinct trapezoids $T_1 = (I'_1, J'_1)$ and $T_2 = (I'_2, J'_2)$ in a family of trapezoids are

disjoint, *i.e.* they contain no common point, if and only if ($I'_1 < I'_2$ and $J'_1 < J'_2$) or ($I'_2 < I'_1$ and $J'_2 < J'_1$) holds. If T_1 and T_2 are indeed disjoint, then one trapezoid is completely to left of the other, say for instance T_1, and this is denoted by $T_1 < T_2$. Finally, a trapezoid graph is an intersection graph of a family of trapezoids.

4.1 Point and unitary 2-interval sets.

We begin our discussion in this section by describing an approximation algorithm for point and unitary 2-interval sets. We call this initial algorithm $\{<, \between\}$-Approx. The general outline of $\{<, \between\}$-Approx consists of the following stages: First $\mathcal{T}(\mathcal{D})$, a family of trapezoids representing the 2-intervals in $\mathcal{D}$, is constructed. Next, the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$ is computed using the algorithm proposed in [9]. Finally, trapezoids in this subset which correspond to non-disjoint 2-intervals in $\mathcal{D}$ are omitted, and the filtered solution is outputted.

Definition 4 (Corresponding trapezoid family). *Let $\mathcal{D}$ be a set of 2-intervals, and let α and β be two distinct horizontal lines which are aligned and such that α is below β. The* corresponding trapezoid family of $\mathcal{D}$, *denoted $\mathcal{T}(\mathcal{D})$, is defined as the family containing a single trapezoid $T = (I', J') \in \mathcal{D}$ for each 2-interval $D = (I, J) \in \mathcal{D}$, where I' is defined over α, J' is defined over β, and $I' = I, J' = J$.*

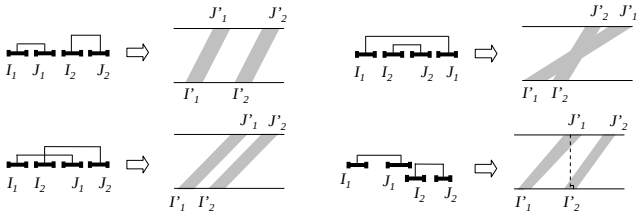

Fig. 7. $\{<, \between\}$-comparable 2-intervals correspond to disjoint trapezoids but the converse is not necessarily true. The bottom right pair of 2-intervals correspond to a pair of clashing trapezoids.

Let $\mathcal{D}$ be a set of 2-intervals and let $\mathcal{T}(\mathcal{D})$ be the corresponding trapezoid family of $\mathcal{D}$. It is not difficult to see that $\{<, \between\}$-comparable 2-intervals in $\mathcal{D}$ correspond to disjoint trapezoids in $\mathcal{T}(\mathcal{D})$, while $\{\sqsubset\}$-comparable 2-intervals in $\mathcal{D}$ correspond to intersecting trapezoids in $\mathcal{T}(\mathcal{D})$ (see Figure 7).

Observation 2. *Any two disjoint 2-intervals in $\mathcal{D}$ are $\{<, \between\}$-comparable if and only if their corresponding trapezoids in $\mathcal{T}(\mathcal{D})$ are disjoint.*

Felsner *et al.* [9] presented an $\mathcal{O}(n \lg n)$ algorithm for finding a maximum disjoint subset in a family of n trapezoids. Unfortunately, this alone does not suffice in our case, since there may be disjoint trapezoids in $\mathcal{T}(\mathcal{D})$ which correspond to non-disjoint 2-intervals in $\mathcal{D}$. (see Figure 7).

Definition 5 (Clashing intervals). *Let $I' = [l(I'), r(I')]$ and $J' = [l(J'), r(J')]$ be two distinct intervals defined over two distinct horizontal lines such that $l(I') \leq l(J')$. The two intervals I' and J' clash, if either $l(I') \leq l(J') \leq r(J') \leq r(I')$ or $l(I') \leq l(J') \leq r(I') \leq r(J')$.*

Definition 6 (Clashing trapezoids). *Let $T_1 = (I'_1, J'_1)$ and $T_2 = (I'_2, J'_2)$ be two distinct trapezoids in a family of trapezoids. The two trapezoids T_1 and T_2 clash, if either I'_1 and J'_2 clash or I'_2 and J'_1 clash.*

Observation 3. *Any pair of 2-intervals in $\mathcal{D}$ are $\{<, \between\}$-comparable if and only if their corresponding trapezoids in $\mathcal{T}(\mathcal{D})$ are disjoint and do not clash.*

Observation 3 is the heart of algorithm $\{<, \between\}$-Approx. Note that the number of maximal (in inclusion order) pairwise disjoint subsets of $\mathcal{T}(\mathcal{D})$ can be exponential, so exhaustively searching through all these for a maximum non-clashing subset is unfeasible. Now, let $\mathcal{T}'$ be the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$. Since the maximum $\{<, \between\}$-comparable subset of 2-intervals $OPT \subseteq \mathcal{D}$ corresponds to a pairwise disjoint non-clashing subset of trapezoids, we have $|OPT| \leq |\mathcal{T}'|$. Next we show that in case $\mathcal{D}$ is a unitary 2-interval set, we can obtain a pairwise non-clashing subset of $\mathcal{T}'$ which is no more than a constant factor smaller than $\mathcal{T}'$, and hence no more than a constant factor smaller than OPT.

Consider the leftmost trapezoid T_0 of $\mathcal{T}'$, and let D_0 be its corresponding 2-interval in $\mathcal{D}$. By definition, any trapezoid in $\mathcal{T}(\mathcal{D})$ has a bottom interval which is completely to the left of its top interval. Hence, T_0 only clashes with trapezoids on its right in $\mathcal{T}'$. Now, if $\mathcal{D}$ is a point 2-interval set, then all 2-intervals with left intervals intersecting the right interval of D_0 have the same left interval, and as $\mathcal{T}'$ is pairwise disjoint, at most one of these has a corresponding trapezoid in $\mathcal{T}'$. Furthermore, if $\mathcal{D}$ is a unitary 2-interval set, distinct intervals involved in $\mathcal{D}$ which are non-disjoint must overlap. Thus, any trapezoid in $\mathcal{T}'$ clashing with T_0 corresponds to a 2-interval with a left interval which contains either endpoints of the right interval of D_0. Since $\mathcal{T}'$ is pairwise disjoint, there can be at most two such trapezoids in $\mathcal{T}'$.

Algorithm $\{<, \between\}$-Approx first computes $\mathcal{T}'$, the maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$, and then repeatedly adds the leftmost trapezoids in $\mathcal{T}'$ to the solution, while omitting all trapezoids which clash with this trapezoid in $\mathcal{T}'$. A schematic description of algorithm $\{<, \between\}$-Approx is given in Figure 8.

Algorithm $\{<, \between\}$-Approx($\mathcal{D}$)

Data : A set of 2-intervals $\mathcal{D}$.
Result : A $\{<, \between\}$-comparable subset of $\mathcal{D}$.
begin
  1. Construct $\mathcal{T}(\mathcal{D})$, the corresponding trapezoid set of $\mathcal{D}$.
  2. Compute $\mathcal{T}' \subseteq \mathcal{T}(\mathcal{D})$, a maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$ using [9].
  3. while $\mathcal{T}' \neq \emptyset$ **do**
    (a) Let T_0 be the leftmost trapezoid in $\mathcal{T}'$.
    (b) Add T_0 to the solution.
    (c) Omit T_0 and all trapezoids clashing with T_0 from $\mathcal{T}'$.
  end
  return *the set of 2-intervals corresponding to the trapezoids in the solution.*
end

Fig. 8. A schematic description of algorithm $\{<, \between\}$-Approx.

Lemma 3. *Algorithm $\{<, \between\}$-Approx is a 3-approximation (2-approximation) algorithm for the* 2-Interval Pattern *problem over the $\{<, \between\}$ model restricted to unitary (point) 2-interval sets.*

Proof. First note that $\{<, \between\}$-Approx outputs a subset of 2-intervals which correspond to pairwise disjoint non-clashing trapezoids. Hence, by Observation 3, this subset is $\{<, \between\}$-comparable. Now,

let OPT be a maximum $\{<, \between\}$-comparable subset of $\mathcal{D}$. Prior to step 3 in the algorithm, we have $|OPT| \leq |\mathcal{T}'|$. Furthermore, if $\mathcal{D}$ is a point 2-interval set, for every trapezoid omitted from $\mathcal{T}'$ in step 3, a trapezoid is added to the solution. Hence $\{<, \between\}$-Approx is a 2-approximation algorithm in this case. The case where $\mathcal{D}$ is unitary is similar, except that here two trapezoids may be omitted for every trapezoid added to the solution. □

Time complexity. Let $|\mathcal{D}| = n$. The family of trapezoids $\mathcal{T}(\mathcal{D})$ can be constructed in $\mathcal{O}(n)$ time, and according to [9], $\mathcal{T}' \subseteq \mathcal{T}(\mathcal{D})$ can be computed in $\mathcal{O}(n \lg n)$ time. Furthermore, if we sort all the right endpoints of intervals involved in $\mathcal{D}$ in an $\mathcal{O}(n \lg n)$ preprocessing stage, we can compute each iteration of step 3 in linear time with respect to the number of trapezoids omitted. As there is only a constant number of such trapezoids in each iteration, step 3 can be computed in $\mathcal{O}(n \lg n)$ time. This gives us a total of $\mathcal{O}(n \lg n)$ running time for the entire algorithm.

4.2 Balanced 2-interval sets.

We next consider balanced 2-interval sets. Bal-$\{<, \between\}$-Approx is a 5-approximation algorithm for this problem. It differs from $\{<, \between\}$-Approx only by the fact that at each iteration of step 3, instead of choosing the leftmost trapezoid in $\mathcal{T}'$ as T_0, we choose the smallest trapezoid (*i.e.* the trapezoid corresponding to the smallest 2-interval) as T_0.

Lemma 4. *Algorithm Bal-$\{<, \between\}$-Approx is a 5-approximation algorithm for the* 2-Interval Pattern *problem over the $\{<, \between\}$ model restricted to balanced 2-interval sets.*

Proof. The correctness of Bal-$\{<, \between\}$-Approx follows again from Observation 3. As for the approximation guarantee, consider $\mathcal{T}'$ at an arbitrary iteration of step 3 in Bal-$\{<, \between\}$-Approx, and let T_0 be the smallest trapezoid of $\mathcal{T}'$ at this iteration. Also let OPT denote the maximum $\{<, \between\}$-comparable subset of $\mathcal{D}$. Since T_0 is the smallest trapezoid, by a similar argument used in Lemma 1, T_0 clashes with at most 4 other trapezoids in $\mathcal{T}'$ at this iteration. Hence, since $|OPT| \leq |\mathcal{T}'|$ prior to step 3, our solution is at least of size $\frac{1}{5}|\mathcal{T}'|$, and the lemma follows. □

Time complexity. Step 3 in Bal-$\{<, \between\}$-Approx can be done in $\mathcal{O}(n \lg n)$ time, where $n = |\mathcal{D}|$, using the same techniques used in Bal-$\{<, \sqsubset, \between\}$-Approx. Hence, as in $\{<, \between\}$-Approx, the entire running time of Bal-$\{<, \between\}$-Approx is $\mathcal{O}(n \lg n)$.

4.3 Unlimited 2-interval sets.

The rest of this section is devoted to the 2-Interval Pattern problem over the $\{<, \between\}$ model for unlimited 2-interval sets. We introduce a slightly more involved modification of $\{<, \between\}$-Approx to obtain a 6-approximation algorithm for unlimited 2-interval sets.

Consider two clashing trapezoids $T_1 = (I'_1, J'_1)$ and $T_2 = (I'_2, J'_2)$ such that $T_1 < T_2$. We say that T_1 *sees* T_2 if either $l(I'_2) \leq l(J'_1) \leq r(I'_2)$ or $l(I'_2) \leq r(J'_1) \leq r(I'_2)$, where $l(J'_1), r(J'_1)$ and $l(I'_2), r(I'_2)$ are the left and right endpoints of J'_1 and I'_2 respectively. Thus, T_1 sees T_2 if one of the endpoints of its top interval is in the range of the bottom interval of T_2 (see Figure 9).

Definition 7 (Nice family of trapezoids). *A family of trapezoids $\mathcal{T}''$ is* nice, *if $\mathcal{T}''$ is pairwise disjoint, and no trapezoid sees any other trapezoid in $\mathcal{T}''$.*

Now, given a pairwise disjoint family of trapezoids $\mathcal{T}'$, computing a nice subset $\mathcal{T}'' \subseteq \mathcal{T}'$ can be done similarly to step 3 in $\{<, \between\}$-Approx. Instead of omitting all trapezoids clashing with the leftmost trapezoid T_0 at every iteration, we omit only those that T_0 sees. Since $\mathcal{T}'$ is pairwise disjoint, T_0 can see at most two trapezoids in $\mathcal{T}'$. Hence, $|\mathcal{T}''| \geq \frac{1}{3}|\mathcal{T}'|$.

Definition 8 (Clashing trapezoid graph). *Given a family $\mathcal{T}$ of trapezoids, the clashing trapezoid graph of $\mathcal{T}$, denoted by $G_{\mathcal{T}}$, is a graph with $\mathcal{T}$ as its vertex set, and two vertices are connected by an edge if and only if their corresponding trapezoids clash.*

Lemma 5. *If $\mathcal{T}''$ is a nice family of trapezoids then $G_{\mathcal{T}''}$ is a forest.*

Proof. Let $\mathcal{T}''$ be a nice family of trapezoids and let $G_{\mathcal{T}''} = (V, E)$ be its corresponding clashing trapezoid graph. Define $G^*_{\mathcal{T}''} = (V^*, E^*)$ as the directed graph obtained by orienting the edges of $G_{\mathcal{T}''}$ according to the precedence relation of $\mathcal{T}''$. In other words, $V^* = V$ and $(T_1, T_2) \in E^*$ if and only if $\{T_1, T_2\} \in E$ and $T_1 < T_2$ in $\mathcal{T}''$. Since $\mathcal{T}''$ is nice, every trapezoid in $\mathcal{T}''$ clashes with at most one trapezoid on its left, and so the in-degree of every vertex $v \in V^*$ is at most one. Hence, any cycle $(v_0, \ldots, v_t, v_0)$ in $G_{\mathcal{T}''}$ is a (directed) cycle in $G^*_{\mathcal{T}''}$. However, in such a case we must have $T_0 < T_t < T_0$, a contradiction. Hence, we conclude that $G_{\mathcal{T}''}$ contains no cycles, and the above lemma holds. □

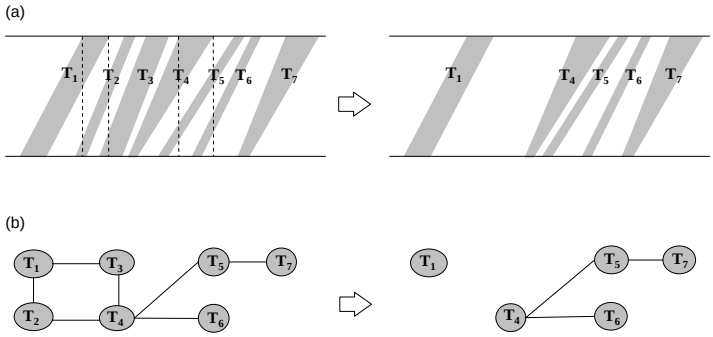

Fig. 9. (a) A family of pairwise disjoint family of trapezoids and a nice subset of this family. Trapezoid T_1 sees both T_2 and T_3 and so these are omitted in order to obtain the nice subset. (b) The corresponding clashing trapezoid graphs of the two families above.

It is well known that the maximum independent set in any forest $G = (V, E)$ is of size at least $\frac{1}{2}|V|$ and that this set can be found in linear time with respect to $|V|$. Also, by definition, since $\mathcal{T}''$ is a pairwise disjoint family of trapezoids, any independent set of $G_{\mathcal{T}''}$ corresponds to a pairwise disjoint non-clashing set of trapezoids, and so it also corresponds to a $\{<, \between\}$-comparable subset of 2-intervals. A schematic description of our algorithm for unlimited 2-intervals sets, called Unl-$\{<, \between\}$-Approx, is given in Figure 10.

Lemma 6. *Algorithm Unl-$\{<, \between\}$-Approx is a 6-approximation algorithm for the* 2-Interval Pattern *problem over the $\{<, \between\}$ model.*

Proof. The correctness of Unl-$\{<, \between\}$-Approx follows from the fact that an independent set in $G_{\mathcal{T}''}$ corresponds to a pairwise disjoint non-clashing subset of trapezoids. Now, let $\mathcal{D}$ be the input set of 2-intervals and let $\mathcal{T}(\mathcal{D})$, $\mathcal{T}'$ and $\mathcal{T}''$ be the trapezoid families as described in the above description of Unl-$\{<, \between\}$-Approx. Also, let OPT be a maximum $\{<, \between\}$-comparable subset of $\mathcal{D}$. We have $|OPT| \leq |\mathcal{T}'|$ and $|\mathcal{T}'| \leq 3|\mathcal{T}''|$. Furthermore, since $G_{\mathcal{T}''}$ is a forest, we have $|V(G_{\mathcal{T}''})| \leq 2\alpha(G_{\mathcal{T}''})$, where $\alpha(G_{\mathcal{T}''})$ is the size of the maximal independent set of $G_{\mathcal{T}''}$. Together we get:

$$|OPT| \leq |\mathcal{T}'| \leq 3|\mathcal{T}''| = 3|V(G_{\mathcal{T}''})| \leq 6\alpha(G_{\mathcal{T}''}),$$

and the lemma follows. □

Algorithm Unl-$\{<, \emptyset\}$-Approx($\mathcal{D}$)

Data : A set of 2-intervals $\mathcal{D}$.
Result : A $\{<, \emptyset\}$-comparable subset of $\mathcal{D}$.
begin
1. Construct $\mathcal{T}(\mathcal{D})$, the corresponding trapezoid set of $\mathcal{D}$.
2. Compute $\mathcal{T}'$, a maximum pairwise disjoint subset of $\mathcal{T}(\mathcal{D})$.
3. Compute $\mathcal{T}''$, a nice subset of $\mathcal{T}'$, such that $|\mathcal{T}''| \geq \frac{1}{3}|\mathcal{T}'|$.
4. Compute $G_{\mathcal{T}''}$ and a maximum independent set I of $G_{\mathcal{T}''}$.
return *the set of 2-intervals corresponding to the maximum independent set I of $G_{\mathcal{T}''}$.*
end

Fig. 10. A schematic description of algorithm Unl-$\{<, \emptyset\}$-Approx.

Time complexity. Let $|\mathcal{D}| = n$. Steps 1-3 in Unl-$\{<, \emptyset\}$-Approx can be computed in $\mathcal{O}(n \lg n)$ time by a similar analysis given in $\{<, \emptyset\}$-Approx. Furthermore, step 4 can be computed in $\mathcal{O}(n)$ time since $G_{\mathcal{T}''}$ is a forest. Hence, the entire algorithm has a total of $\mathcal{O}(n \lg n)$ running time.

5 Approximation framework for weighted 2-interval sets

In this section we consider the weighted version of the 2-Interval Pattern problem.

Definition 9. *Let $\mathcal{D}$ be a set of 2-intervals and let $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, $\mathcal{R} \neq \emptyset$, be a given model. Also let $w : \mathcal{D} \to \mathbb{R}$ be a weight function. The* Weighted 2-Interval Pattern *problem asks to find a maximum weight $\mathcal{R}$-comparable subset of $\mathcal{D}$.*

All algorithms for the polynomial solvable models of 2-Interval Pattern given in [6, 20] apply to the weighted version as well. In the following we show that our results also extend to Weighted 2-Interval Pattern.

We denote $w(\mathcal{D}')$ as the sum $\Sigma_{D \in \mathcal{D}'} w(D)$, for any subset of 2-intervals $\mathcal{D}' \subseteq \mathcal{D}$ and any weight function $w : \mathcal{D} \to \mathbb{R}$. For a model $\mathcal{R}$ and a weight function w, a given subset $\mathcal{S} \subseteq \mathcal{D}$ is *r-approximate* with respect to $\mathcal{R}$ and w, if $w(\mathcal{S}) \geq \frac{1}{r} w(\mathcal{D}')$ for any $\mathcal{R}$-comparable $\mathcal{D}' \subseteq \mathcal{D}$.

5.1 A local ratio approximation framework

The local-ratio technique [2–4] is based on the Local-Ratio Theorem, which in our case is stated as follows.

Theorem 4 (Local-Ratio [2]). *Let $\mathcal{R}$ be a given model and let w, w_1, and w_2 be weight functions such that $w = w_1 + w_2$. Then, if $\mathcal{D}' \subseteq \mathcal{D}$ is r-approximate, both with respect to $(\mathcal{R}, w_1)$, and with respect to $(\mathcal{R}, w_2)$, then $\mathcal{D}'$ is also r-approximate with respect to $(\mathcal{R}, w)$.*

In Figure 11, we present a local ratio approximation framework that is based on the approximation framework for scheduling and resource allocation from [2]. It uses the following definition: Given a set of 2-intervals $\mathcal{D}$, a model $\mathcal{R}$, and a 2-interval $D \in \mathcal{D}$, $N[D]$ denotes the subset of 2-intervals in $\mathcal{D}$ that are not $\mathcal{R}$-comparable with D ($D \in N[D]$). We assume that the initial weights are positive (2-intervals with non-positive weights can be omitted). However, note that the weights may become negative during the execution of the algorithm.

The selection of D_0 determines the approximation ratio of the algorithm. Informally, the approximation ratio would be small, if in each iteration, we are able to choose a 2-interval D_0 such that the intersection of $N[D_0]$ with any $\mathcal{R}$-comparable subset in this iteration is small.

Algorithm LR($\mathcal{D}, \mathcal{R}, w$)

Data : A set of 2-intervals $\mathcal{D}$, a model $\mathcal{R}$, and a weight function w.
Result : An $\mathcal{R}$-comparable subset of $\mathcal{D}$.
begin
1. if $\mathcal{D} = \emptyset$ **then return** $\emptyset$.
2. Select a 2-interval $D_0 \in \mathcal{D}$.
3. Define $w_1(D) = \begin{cases} w(D_0) & D \in N[D_0], \\ 0 & \text{otherwise} \end{cases}$.
4. Define $w_2 = w - w_1$.
5. $\mathcal{D}^+ \leftarrow \{D \in \mathcal{D} : w_2(D) > 0\}$.
6. $\mathcal{S} \leftarrow \text{LR}(\mathcal{D}^+, \mathcal{R}, w_2)$.
7. if $\mathcal{S} \cup \{D_0\}$ is $\mathcal{R}$-comparable **then** $\mathcal{S} \leftarrow \mathcal{S} \cup \{D_0\}$.
return $\mathcal{S}$.
end

Fig. 11. A local ratio approximation framework.

Definition 10 (D_0-maximal subset). *Let $\mathcal{R}$ be a model, and let D_0 be a 2-interval. We say that an $\mathcal{R}$-comparable set $\mathcal{S}$ is* D_0-maximal *if either $D_0 \in \mathcal{S}$, or $D_0 \notin \mathcal{S}$ but $\mathcal{S} \cup \{D_0\}$ is not $\mathcal{R}$-comparable.*

Definition 11 (r-effective weight function). *Given a model $\mathcal{R}$ and a 2-interval D_0, a weight function w_1 is called* r-effective *with respect to D_0, if every D_0-maximal $\mathcal{R}$-comparable subset $\mathcal{S} \subseteq \mathcal{D}$ is r-approximate with respect to w_1 and $\mathcal{R}$.*

Lemma 7. *If w_1 is r-effective with respect to D_0 in every recursive call of algorithm LR, then LR computes an r-approximate $\mathcal{R}$-comparable subset $\mathcal{S}$.*

Proof. First, the solution computed by the algorithm is $\mathcal{R}$-comparable by construction. We prove it is r-approximate by induction on the number of recursive calls (which is bounded by n). At the recursive basis, the solution $\mathcal{S}$ returned is the empty set, and hence it is optimal and clearly r-approximate. For the inductive step, assume that at some recursive call of the algorithm, the intermediate solution $\mathcal{S}$ computed at step 6, is r-approximate with respect to w_2. Step 7 ensures that $\mathcal{S}$ is D_0-maximal, and so $\mathcal{S}$ is r-approximate with respect to w_1 after this step. Furthermore, since $w_2(D_0) = 0$, $\mathcal{S}$ remains r-approximate with respect to w_2 after this step as well. Therefore, by the Local Ratio Theorem, we get that the solution returned at the end of this recursive call is r-approximate with respect to w, and the lemma follows. □

Next, we give an alternative analysis for algorithm LR. Basically, we show that the approximation ratio would be small, if in each iteration we are able to choose a 2-interval D_0 such that $N[D_0]$ is small. As we shall later see, this will be useful when the input set of 2-intervals for algorithm LR is not the original set $\mathcal{D}$, but rather a subset whose weight is at least the weight of a maximum weight $\mathcal{R}$-comparable subset of $\mathcal{D}$.

Lemma 8. *If $|N[D_0]| \leq r$ in every recursive call of algorithm LR, then LR computes an r-approximate $\mathcal{R}$-comparable subset $\mathcal{S}$.*

Proof. Let w_1^i and D_0^i be the weight w_1 and the 2-interval D_0 in the ith recursive call of Algorithm LR, respectfully. First observe that $\sum_i w_1^i(D) \geq w(D)$ for any 2-interval $D \in \mathcal{D}$. Hence,

$$\sum_{D \in \mathcal{D}} \sum_i w_1^i(D) \geq \sum_{D \in \mathcal{D}} w(D) = w(\mathcal{D}).$$

On the other hand, $w(D) = \sum_i w_1^i(D)$ for any $D \in \mathcal{S}$, and therefore

$$w(\mathcal{S}) = \sum_{D \in \mathcal{S}} \sum_i D = \sum_i w_1^i(\mathcal{S}).$$

Now, in any recursive call i, $\mathcal{S}$ is D_0-maximal, and so $w_1^i(\mathcal{S}) \geq w_1(D_0^i)$. Furthermore, by definition of w_1^i and since $|N[D_0^i]| \leq r$, we have $w_1^i(D_0^i) \geq \frac{1}{r} \sum_{D \in \mathcal{D}} w_1^i(D)$. Accumulating all these inequalities together we get:

$$w(\mathcal{S}) = \sum_i w_1^i(\mathcal{S}) \geq \sum_i w_1^i(D_0^i) \geq \frac{1}{r} \sum_i \sum_{D \in \mathcal{D}} w_1^i(D) \geq \frac{1}{r} w(\mathcal{D}) \ ,$$

and we are done. □

We now turn to show that the analysis of algorithm LR given in Lemma 7 and 8 is sufficient for extending our results from the previous sections to the Weighted 2-Interval Pattern problem. More specifically, we show that the algorithms for the unweighted version of the problem can be extended to the weighted version while still maintaining their approximation factors.

5.2 The $\{<, \sqsubset, \between\}$ model

First, both the 4-approximation algorithm for unlimited 2-interval sets and the 3-approximation algorithm for unitary 2-interval sets from [5] work for weighted instances.

Lemma 9. *There is a 4-approximation algorithm for the* Weighted 2-Interval Pattern *problem over the* $\{<, \sqsubset, \between\}$ *model restricted to balanced 2-interval sets.*

Proof. We show a 4-approximation algorithm for weighted balanced 2-interval sets. Our algorithm uses the approximation framework of algorithm LR by selecting D_0 as the smallest 2-interval in $\mathcal{D}$. Due to Lemma 7, to show that this algorithm has an approximation factor of 4, it is enough to show that w_1 is 4-effective with respect to D_0 in every recursive call of LR.

Consider a D_0-maximal $\mathcal{R}$-comparable subset $\mathcal{S}$ at any recursive call of algorithm LR. Since D_0 is the smallest 2-interval in $\mathcal{D}$, no interval is properly contained in the left or right interval of D_0. Hence, for any $\{<, \sqsubset, \between\}$-comparable $\mathcal{D}' \subseteq \mathcal{D}$, at most four 2-intervals in $\mathcal{D}'$ are also in $N[D_0]$. As only the 2-intervals in $N[D_0]$ are assigned a positive weight ($w_1(D_0)$) by w_1, we have $w_1(\mathcal{D}') \leq 4w_1(D_0)$. On the other hand, we have $w_1(\mathcal{S}) \geq w_1(D_0)$, since $\mathcal{S}$ is D_0-maximal and $\mathcal{S} \cap N[D_0] \neq \emptyset$. Therefore $w_1(\mathcal{D}') \leq 4w_1(\mathcal{S})$ and so w_1 is 4-effective with respect to D_0. □

5.3 The $\{\sqsubset, \between\}$ model

The following is the weighted variant of Observation 1.

Observation 5. *Let OPT denote the maximum weight* $\{\sqsubset, \between\}$*-comparable subset of* $\mathcal{D}$*. Then OPT is a pairwise disjoint subset of a set of* 2*-intervals* $\mathcal{D}'$ *(*$OPT \subseteq \mathcal{D}' \subseteq \mathcal{D}$*), such that* $\mathcal{C}(\mathcal{D}')$*, the covering intervals of* $\mathcal{D}$*, corresponds to a maximal clique in* $\Omega_{\mathcal{C}(\mathcal{D})}$*, the interval graph of all covering intervals of* $\mathcal{D}$*.*

Hence, our algorithm for Weighted 2-Interval Pattern over $\{\sqsubset, \between\}$ is very much similar to $\{\sqsubset, \between\}$-Approx (Figure 5). Here, we search through all maximal cliques of $\Omega_{\mathcal{C}(\mathcal{D})}$ for an approximation of the maximum weight $\{\sqsubset, \between\}$-comparable solution. This is done using the algorithms given above (Section 5.2).

Corollary 2. *There is a 4-approximation* (3*-approximation*) *algorithm for the* Weighted 2-Interval Pattern *problem over the* $\{\sqsubset, \between\}$ *model restricted to unlimited and balanced* (*unitary*) *2-interval sets.*

5.4 The $\{<, \between\}$ model

We begin by considering the case of unitary and point 2-intervals. Recall algorithm $\{<, \between\}$-Approx. In Steps 1 and 2, we compute the corresponding trapezoid family $\mathcal{T}(\mathcal{D})$ of $\mathcal{D}$ and the maximum pairwise disjoint subset $\mathcal{T}'$ of this family. Let $\mathcal{D}'$ be the set of trapezoids corresponding to $\mathcal{T}'$. We modify $\{<, \between\}$-Approx by replacing step 3 in the algorithm with a call to LR($\mathcal{D}', \{<, \between\}, w$) (Figure 11), where D_0 is selected as the 2-interval which corresponds to the leftmost trapezoid at each recursive call. The modified version of $\{<, \between\}$-Approx then outputs the solution given by algorithm LR. In Section 4.1 we showed that T_0, the trapezoid corresponding to D_0, clashes with at most two trapezoids in $\mathcal{T}'$ in case $\mathcal{D}$ is unitary, and at most one trapezoid in case $\mathcal{D}$ is a point 2-interval set. Hence, $|N[D_0]| \leq 3$ in every recursive call of LR if $\mathcal{D}$ is unitary, and $|N[D_0]| \leq 2$ in case $\mathcal{D}$ is a point 2-interval set. Therefore, by Lemma 8, algorithm LR computes a 3-approximate solution for unitary 2-interval sets and a 2-approximate solution for point 2-interval sets.

The case of balanced 2-intervals is similar, except that here we select D_0 as the smallest 2-interval in every recursive call of algorithm LR. As $|N[D_0]| \leq 5$ in every recursive call, LR computes a 5-approximate solution in this case.

For unlimited 2-interval sets, we modify algorithm Unl-$\{<, \between\}$-Approx (Figure 10) by replacing step 3 with a variant of algorithm LR. In this variant, we choose D_0 to be the 2-interval corresponding to the leftmost trapezoid $T_0 \in \mathcal{T}'$. Next, we replace $N[D_0]$ by $N'[D_0]$, where $N'[D_0]$ is the set of all 2-intervals which correspond to trapezoids that T_0 sees. Finally, instead of requiring $\mathcal{S}$ to be $\mathcal{R}$-comparable, we require the corresponding trapezoid family be nice. By the analysis given in Section 4.3, we have $|N'[D_0]| \leq 3$ in any recursive call of algorithm LR, and so by Lemma 8, this variant of algorithm LR computes a nice trapezoid family $\mathcal{T}''$ of size at least $\frac{1}{3}|\mathcal{T}'|$. From here the analysis of this algorithm is similar to the unweighted case.

Corollary 3. *There is a 6-approximation* (*5-approximation, 3-approximation, and 2-approximation*) *algorithm for the* Weighted 2-Interval Pattern *problem over the* $\{<, \between\}$ *model restricted to unlimited* (*balanced, unitary, and point*) *2-interval sets.*

6 Conclusions and future work

In this paper we addressed the problem of approximating the 2-Interval Pattern problem over its various models and restrictions. We presented algorithms with constant approximation factor guarantees for all **NP**-hard cases of the problem. In addition, we showed that these can be extended to the weighted version of the problem with no cost to the approximation factors.

A first natural open problem to consider is improving the approximation factors of our algorithms. An additional problem is to provide an efficient algorithm for the $\{<, \sqsubset, \between\}$ model with unlimited 2-interval sets. This is of great interest, since the inefficiency of the algorithm in [5] also propagates to our suggested algorithm for the $\{\sqsubset, \between\}$ model. Note that [5] give a fast $\mathcal{O}(\lg|\mathcal{D}|)$-approximation algorithm for this case.

As for hardness of approximation results, unlike the $\{<, \sqsubset, \between\}$ and $\{\sqsubset, \between\}$ models, the **APX**-hardness results described in [5] do not extend easily to the $\{<, \between\}$ model. Whether the 2-Interval Pattern problem over the $\{<, \between\}$ model is **APX**-hard, and if so, for what restrictions on the input does this still hold, remains open.

Finally, and perhaps most interesting, is to determine the time complexity of the 2-Interval Pattern problem over the $\{<, \between\}$ model restricted to point 2-intervals. This has been posed as an open problem both in [20] and in [6], and is still left open by this paper.

Acknowledgments. The authors would like to express their gratitude to Martin C. Golumbic for fruitful discussions and valuable remarks.

References

1. T. Akutsu. Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Applied Mathematics*, 104:45-62, 2000.
2. A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, and B. Shieber. A unified approach to approximating resource allocation and schedualing. *Journal of the ACM*, 48(5):1069–1090, 2001.
3. R. Bar-Yehuda. One for the price of two: A unified approach for approximating covering problems. *Algorithmica*, 27(2):131–144, 2000.
4. R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–46, 1985.
5. R. Bar-Yehuda, M.M. Halldorsson, J. Naor, H. Shachnai and I. Shapira. Scheduling split intervals. *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2002)*, 732-741.
6. G. Blin, G. Fertin and S. Vialette. New results for the 2-interval pattern problem. *Proceedings of the 15th Annual Symposium on Combinatorial Pattern Matching (CPM 2004)*, Lecture Notes in Computer Science 3109, Springer-Verlag, 311-322.
7. I. Dagan, M.C. Golumbic and R.Y. Pinter. Trapezoid graphs and their coloring. *Discrete Applied Mathematics*, 21:35-46, 1988.
8. P. Evans. Finding common subsequences with arcs and pseudoknots. *Proceedings of the 10th Annual Symposium on Combinatorial Pattern Matching (CPM 1999)*, Lecture Notes in Computer Science 1645, Springer-Verlag, 270-280.
9. S. Felsner, R. Müller and L. Wernisch. Trapezoid graphs and generalizations: Geometry and algorithms. *Discrete Applied Mathematics*, 74:13-32, 1997.
10. F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1:180-187, 1972.
11. D. Goldman, S. Istrail and C.H. Papadimitriou. Algorithmic aspects of protein structure similarity. *Proceedings of the 40th Annual Symposium of Foundations of Computer Science (FOCS 1999)*, 512-522.
12. M.C. Golumbic. *Algorithmic graph theory and perfect graphs.* Academic Press, New York, 1980.
13. J. Gramm, J. Guo and R. Niedermeier. Pattern matching for arc-annotated sequences. *Proceedings of the 22th Conference on Foundations of Software Technology and Theoretical Computer Science (FCTTSC 1999)*, 182-193.
14. J.R. Griggs, and D.B. West. Extremal values of the interval number of a graph. *SIAM Journal of Algebraic and Discrete Methods*, 1:1-7, 1979.
15. S. Ieong, M.Y. Kao, T.W. Lam, W.K. Sung and S.M. Yiu. Predicting RNA secondary structures with arbitrary pseudoknots by maximizing the number of stacking pairs. *Proceedings of the 2nd Symposium on Bioinformatics and Bioengineering (BIBE 2002)*, 183-190.
16. R.B. Lyngsø, C.N.S. Pedersen. RNA pseudoknot prediction in energy based models. *Journal of Computational Biology*, 7:409-428, 2000.
17. T.A. McKee, F.R. McMorris. *Topics in intersection graph theory.* SIAM monographs on discrete mathematics and applications, 1999.
18. P.B. Moore. Structural motifs in RNA. *Annual review of Biochemistry.*, 68:287-300, 1999.
19. W.T. Trotter, and F. Harary. On double and multiple interval graphs. *Journal of Graph Theory*, 3:205-211, 1979.
20. S. Vialette. On the computational complexity of 2-interval pattern matching problems. *Theoretical Computer Science*, 312:335-379, 2004.
21. M.S. Waterman. *Introduction to computational biology. Maps, sequences and genomes.* Chapman and Hall, London, 1995.