



Finding common structured patterns in linear graphs[☆]

Guillaume Fertin^a, Danny Hermelin^b, Romeo Rizzi^c, Stéphane Vialette^{d,*}

^a LINA, CNRS UMR 6241, Université de Nantes, 2 rue de la Houssinière, 44322 Nantes, France

^b Department of Computer Science, University of Haifa, Mount Carmel, Haifa 31905, Israel

^c DIMI, Università degli Studi di Udine, Via delle Scienze, 208 I-33100 Udine (UD), Italy

^d LIGM, CNRS UMR 8049, Université Paris-Est Marne-la-Vallée, 5 Bd Descartes 77454 Marne-la-Vallée, France

ARTICLE INFO

Article history:

Received 13 November 2007

Received in revised form 12 February 2010

Accepted 15 February 2010

Communicated by A. Apostolico

Keywords:

Linear graphs

Approximation

ABSTRACT

A linear graph is a graph whose vertices are linearly ordered. This linear ordering allows pairs of disjoint edges to be either preceding ($<$), nesting (\sqsubset) or crossing (\bowtie). Given a family of linear graphs, and a non-empty subset $\mathcal{R} \subseteq \{<, \sqsubset, \bowtie\}$, we are interested in the MAXIMUM COMMON STRUCTURED PATTERN (MCSP) problem: find a maximum size edge-disjoint graph, with edge pairs all comparable by one of the relations in \mathcal{R} , that occurs as a subgraph in each of the linear graphs of the family. The MCSP problem generalizes many structure-comparison and structure-prediction problems that arise in computational molecular biology.

We give tight hardness results for the MCSP problem for $\{<, \bowtie\}$ -structured patterns and $\{\sqsubset, \bowtie\}$ -structured patterns. Furthermore, we prove that the problem is approximable within ratios: (i) $2\mathcal{H}(k)$ for $\{<, \bowtie\}$ -structured patterns, (ii) $k^{1/2}$ for $\{\sqsubset, \bowtie\}$ -structured patterns, and (iii) $O(\sqrt{k \log k})$ for $\{<, \sqsubset, \bowtie\}$ -structured patterns, where k is the size of the optimal solution and $\mathcal{H}(k) = \sum_{i=1}^k 1/i$ is the k th harmonic number. Also, we provide combinatorial results concerning different types of structured patterns that are of independent interest in their own right.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Many biological molecules such as RNA and proteins exhibit a three-dimensional structure that determines most of their functionalities. This three-dimensional structure can be modeled in two dimensions by an edge-disjoint linear graph, *i.e.*, a graph with linearly ordered vertices that are incident to exactly one edge. The corresponding structure-similarity or structure-prediction problems that arise in such contexts usually translate to finding common edge-disjoint subgraphs, or common *structured patterns*, that occur in a family of general linear graphs. Examples of such problems are the LONGEST COMMON SUBSEQUENCE [22,23] problem, the MAXIMUM COMMON ORDERED TREE INCLUSION [2,9,24] problem, the ARC-PRESERVING SUBSEQUENCE [4,17,20] problem, and the MAXIMUM CONTACT MAP OVERLAP [18] problem. In this paper, we study a general framework for such problems which we call the MAXIMUM COMMON STRUCTURED PATTERN (MCSP) problem.

The MCSP problem was originally introduced (under a different name) by Davydov and Batzoglou [11] in the context of (non-coding) RNA secondary structure prediction via multiple structural alignment. There, an RNA sequence of n nucleotides is represented by a linear graph with n vertices, and an edge connects two vertices if and only if their corresponding

[☆] A preliminary version of the paper appeared in Combinatorial Pattern Matching, 18th Annual Symposium, CPM 2007, London, Canada, July 9–11, 2007, Proceedings. Lecture Notes in Computer Science, 4580, Springer, 2007, ISBN 978-3-540-73436-9.

* Corresponding author. Tel.: +33 1 60 95 77 49; fax: +33 1 60 95 75 57.

E-mail addresses: Guillaume.Fertin@lina.univ-nantes.fr (G. Fertin), danny@cri.haifa.ac.il (D. Hermelin), Romeo.Rizzi@dimi.uniud.it (R. Rizzi), viallette@univ-mlv.fr (S. Vialette).

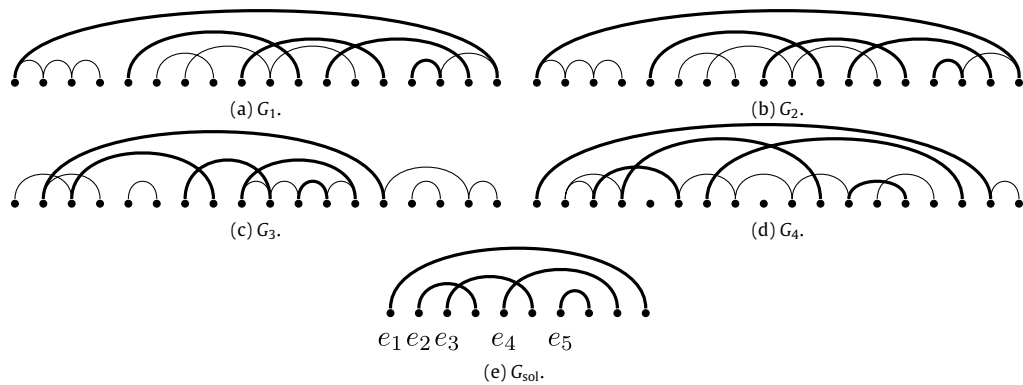


Fig. 1. Four linear graphs G_1 , G_2 , G_3 and G_4 and a $\{<, \sqsubset, \emptyset\}$ -comparable common structured pattern (depicted also as G_{sol} in the bottom part). The occurrence of the structured pattern G_{sol} in each graph is emphasized in bold. Edges e_2 , e_3 , e_4 and e_5 are nested in edge e_1 ; edges e_2 and e_3 precede edge e_5 ; edge e_2 precedes edge e_4 and crosses edge e_3 , while edge e_3 crosses both edges e_2 and e_4 .

nucleotides are complementary. A family of linear graphs is then used to represent a family of functionally-related RNAs, and a common structured pattern in such a family is considered to be a putative common secondary structure element of the family.

The ordering amongst the vertices of a linear graph allows a pair of disjoint edges in the graph to be either preceding ($<$), nesting (\sqsubset), or crossing (\emptyset). Since most RNA secondary structures translate to linear graphs with non-crossing edges, Davydov and Batzoglou [11] focused on the variant of the MCSP problem where the common structured pattern is required to be non-crossing. In other words, they focus on finding maximum common $\{<, \sqsubset\}$ -structured patterns. However, there are known RNAs which have secondary structures that translate to linear graphs with a few edge-crossings (pseudo-knotted RNA secondary structures). Also, when predicting proteins rather than RNA structures, the non-crossing restriction becomes an even bigger limitation since the folding structures of proteins are often more complex than those of RNAs. In [19], it is argued that many important protein secondary structure elements like alpha helices and anti-parallel beta sheets exhibit $\{<, \emptyset\}$ -structured patterns, *i.e.*, patterns which are non-nesting rather than non-crossing.

In the following, we present a framework which extends the work of Davydov and Batzoglou [11] by considering different types of common structured patterns. Following [35], we consider structured patterns that are allowed to have crossing edges, and which might also be restricted to be non-nesting or non-preceding. More specifically, the MCSP problem receives as input a family of linear graphs and a non-empty subset $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, and the goal is to find a maximum common \mathcal{R} -structured pattern (see Fig. 1). We study the combinatorics behind the structures of these different types of patterns, with a focus on approximation algorithms for the MCSP problem.

The paper is organized as follows. In the remaining part of this section we briefly review related work and notations that will be used throughout the paper. In Section 2, we discuss simple structured patterns, *i.e.*, \mathcal{R} -structured patterns, where $\mathcal{R} \in \{<, \sqsubset, \emptyset\}$, and $\{<, \sqsubset\}$ -structured patterns. Following this, we discuss the more complex $\{<, \emptyset\}$ -structured patterns and $\{\sqsubset, \emptyset\}$ -structured patterns in Sections 3 and 4, respectively. In Section 5, we deal with general structured patterns, *i.e.*, $\{<, \sqsubset, \emptyset\}$ -structured patterns. An overview of the paper, along with some open problems, is given in Section 6.

1.1. Related work

There are many structural comparison problems that are closely related to the MCSP problem. First, as mentioned previously, the MCSP problem for $\{<, \sqsubset\}$ -structured patterns has been studied by Davydov and Batzoglou in [11] (coined as the MAXIMUM COMMON NESTED SUBGRAPH problem). Recently, new results concerning this problem appeared in [28]. We discuss the results of both these works in Section 2. Below we list other related problems.

Closely related to the MCSP problem are the ARC-PRESERVING SUBSEQUENCE [4,17,20], and MAXIMUM CONTACT MAP OVERLAP [18] problems. Both are concerned with finding maximum common subgraphs in a pair of linear graphs, except that in the ARC-PRESERVING SUBSEQUENCE problem the vertices of the linear graphs are assigned letters from some given alphabet, and an occurrence of a common subgraph in each of the linear graphs is required to preserve the letters, as well as their arc structure. Another closely related problem is the PATTERN MATCHING OVER 2-INTERVAL SET problem [35], where one asks whether a given structured pattern occurs in a given 2-interval set, which is a generalization of a linear graph. The 2-INTERVAL PATTERN problem [5,10,35] asks to find the maximum \mathcal{R} -structured pattern, for some given $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, in a single family of 2-interval sets. Also, note that an extensive literature is devoted to combinatorial investigations of complete matchings of $\{1, \dots, 2n\}$ [33,8].

There is a well-known bijective correspondence between $\{<, \sqsubset\}$ -structured patterns and ordered forests – the nesting relation corresponds to the ancestor/predecessor relationship between the nodes, and the precedence relation corresponds to their order. Hence, the MCSP problem for $\{<, \sqsubset\}$ -structured patterns can be viewed as the problem of finding a tree which is included in all trees of a given tree family, the MAXIMUM COMMON ORDERED TREE INCLUSION problem. Determining

whether a tree is included in another is studied in [2,9,24]. Finding the maximum common tree included in a pair of trees can be done using the algorithms given in [12,25,32]. The MCSP problem for $\{<, \sqsubset\}$ -structured patterns has been studied in [11,28]. We discuss the results there in Section 2.

Like $\{<, \sqsubset\}$ -structured patterns, $\{\sqsubset, \emptyset\}$ -structured patterns also correspond to natural combinatorial objects, namely permutations (see Section 4). In [7], the authors studied the problem of determining whether a permutation occurs in another permutation as a pattern, the so called PATTERN MATCHING FOR PERMUTATIONS problem. This problem corresponds to determining whether a $\{\sqsubset, \emptyset\}$ -structured pattern is a subpattern of another $\{\sqsubset, \emptyset\}$ -structured pattern. Bose, Buss, and Lubiw proved that PATTERN MATCHING FOR PERMUTATIONS is **NP**-complete [7].

Determining whether a given $\{<, \emptyset\}$ -structured pattern occurs in a general linear graph has been studied in [19,29]. Gramm [19] gave a polynomial-time algorithm for this problem. Recently, Li and Li [29] proved that this algorithm was incorrect and showed the problem was in fact **NP**-complete. Prior to this, Blin et al. [5] proved that a generalization of this problem, where the linear graph is replaced by a 2-interval set, is **NP**-complete.

Finally, probably the oldest and most famous problem related to the MCSP problem is the LONGEST COMMON SUBSEQUENCE (LCS) problem [22,23], where one wishes to find the longest common subsequence in two or more sequences. Important developments of the initial algorithms of [22,23] can be found in [3,15,31]. Maier [30] proved that the LCS problem for multiple sequences is **NP**-hard.

1.2. Terminology and basic definitions

For a graph G , we denote $V(G)$ as the set of vertices and $E(G)$ as the set of edges. The *order* and the *size* of G stand for $|V(G)|$ and $|E(G)|$, respectively. A *linear graph* of order n is a vertex-labeled graph where each vertex is labeled by a distinct label from $\{1, 2, \dots, n\}$. Thus, it can be viewed as a graph with vertices embedded on the integral line, yielding a total order amongst them. In case of linear graphs, we write an edge between vertices i and j , $i < j$, as the pair (i, j) . Two edges of a linear graph are *disjoint* if they do not share a common vertex. A linear graph G is said to be *edge-disjoint* if it is composed of disjoint edges, i.e., G is a matching. Of particular interest are the relations between pairs of disjoint edges [35]. Let $e = (i, j)$ and $e' = (i', j')$ be two disjoint edges in a linear graph G ; we write:

- $e < e'$ (e precedes e') if $i < j < i' < j'$,
- $e \sqsubset e'$ (e is nested in e') if $i' < i < j < j'$, and
- $e \emptyset e'$ (e and e' cross) if $i < i' < j < j'$.

Two edges e and e' are R -comparable, for some $R \in \{<, \sqsubset, \emptyset\}$, if eRe' or $e'Re$. For a subset $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, $\mathcal{R} \neq \emptyset$, e and e' are said to be \mathcal{R} -comparable if e and e' are R -comparable for some $R \in \mathcal{R}$. A set of edges E (or a linear graph G with $E(G) = E$) is \mathcal{R} -comparable if any pair of distinct edges $e, e' \in E$ are \mathcal{R} -comparable. A *subgraph* of a linear graph G is a linear graph H which can be obtained from G by a series of vertex and edge deletions, where a deletion of vertex i results in removing vertex i and all edges incident to it from the graph, and then relabeling all vertices j with $j > i$ to $j - 1$. An edge-disjoint subgraph of a linear graph is called a *structured pattern*. For a vertex subset $V \subseteq V(G)$ (resp. edge subset $E \subseteq E(G)$), we let $G[V]$ (resp. $G[E]$) denote the subgraph induced by V (resp. E), i.e. the subgraph obtained by deleting all vertices $V(G) \setminus V$ (resp. all edges $E(G) \setminus E$). By convention, we let $G[i \dots j]$, $1 \leq i \leq j \leq |V(G)|$, denote the subgraph induced by all vertices labeled k with $i \leq k \leq j$. For a family of linear graphs $\mathcal{G} = G_1, \dots, G_n$, a *common structured pattern* of \mathcal{G} is an edge-disjoint linear graph H that is a subgraph of G_i , for all $1 \leq i \leq n$. Following the above notation, H is called an \mathcal{R} -structured pattern, for some non-empty $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, if $E(H)$ is \mathcal{R} -comparable. We are now in position to formally define the MCSP problem.

Definition 1. Given a family of linear graphs $\mathcal{G} = G_1, \dots, G_n$ and a subset $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, $\mathcal{R} \neq \emptyset$, the MAXIMUM COMMON STRUCTURED PATTERN (MCSP) problem asks to find a maximum-size common \mathcal{R} -structured pattern of \mathcal{G} .

We will use the following terminology to describe special edge-disjoint linear graphs. A linear graph is called a *sequence* if it is $\{<\}$ -comparable, a *tower* if it is $\{\sqsubset\}$ -comparable, and a *staircase* if it is $\{\emptyset\}$ -comparable. We define the *width* (resp. *height* and *depth*) of a linear graph to be the size of the maximum cardinality sequence (resp. tower and staircase) subgraph of the graph. A $\{<, \sqsubset\}$ -comparable linear graph with the additional property that any two maximal towers in it do not share an edge is called a *sequence of towers*. Similarly, a $\{<, \emptyset\}$ -comparable linear graph is a *sequence of staircases* if any two maximal staircases do not share an edge. A *tower of staircases* is a $\{\sqsubset, \emptyset\}$ -comparable linear graph where any pair of maximal staircases do not share an edge, and a *staircase of towers* is a $\{\sqsubset, \emptyset\}$ -comparable linear graph where any pair of maximal towers do not share an edge. A sequence of towers (resp. sequence of staircases, tower of staircases, and staircase of towers) is *balanced* if all of its maximal towers (resp. staircases, staircases, and towers) are of equal size. Fig. 2 illustrates an example of the above types of linear graphs.

1.3. Our results

The results presented in this paper are as follows: We give tight hardness results for the MCSP problem for $\{<, \emptyset\}$ -structured patterns and $\{\sqsubset, \emptyset\}$ -structured patterns, extending the hardness result given in [28] for $\{<, \sqsubset\}$ -structured patterns. Furthermore, we provide approximation algorithms that prove that the problem is approximable within ratios:

- $2\mathcal{H}(k)$ for $\{<, \emptyset\}$ -structured patterns,

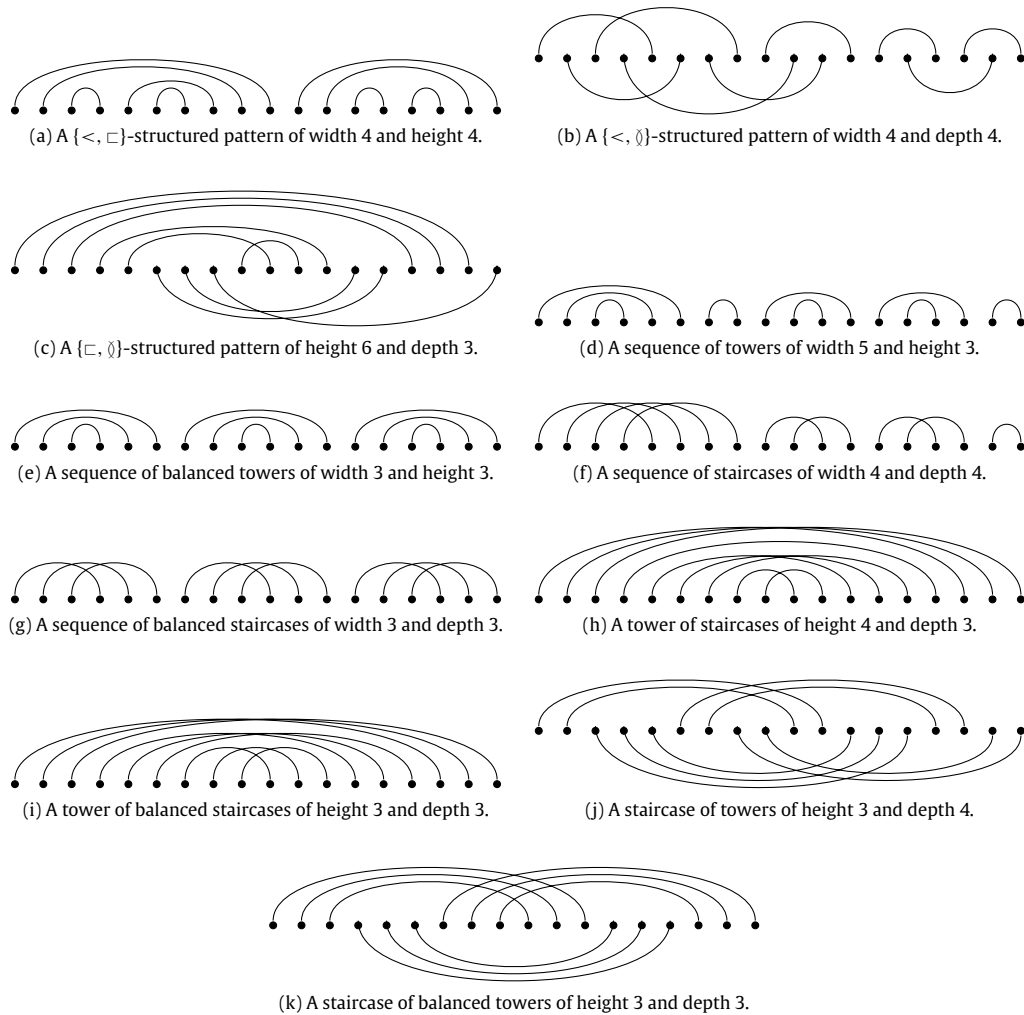


Fig. 2. Some restricted structured patterns. Edges are drawn above or below the vertices with no particular signification.

- $k^{1/2}$ for $\{\square, \emptyset\}$ -structured patterns, and
- $O(\sqrt{k \log k})$ for $\{<, \square, \emptyset\}$ -structured patterns,

where k denotes the size of an optimal solution, and $\mathcal{H}(k) = \sum_{i=1}^k 1/i$ is the k th harmonic number. The respective running times of these algorithms are $O(nm^{3.5} \log m)$, $O(nm^{1.5})$, and $O(nm^{3.5} \log m)$, where n denotes the number of input linear graphs, and m denotes the maximum number of edges in any graph of the input. Along the way, we provide some combinatorial results concerning the different types of structured patterns that are of independent interest in their own right.

2. Simple and $\{<, \square\}$ -structured patterns

We begin our study by considering the MCSP problem for simple structured patterns (R -structured pattern for a single relation $R \in \{<, \square, \emptyset\}$), and for $\{<, \square\}$ -structured patterns. We first discuss the analogy between the relations we defined for disjoint edges in a linear graph and well-studied relations defined for families of intervals. We show that known algorithms on interval families can be used to solve the MCSP problem for simple structured patterns in polynomial-time. Following this, we discuss results presented in [11,28] for the MCSP problem for $\{<, \square\}$ -structured patterns. Most of the results in this section are known, and are presented here for sake of completeness.

For a given linear graph G of size m , let $\mathcal{I}(G) = \{[i, j] \mid (i, j) \in E(G)\}$ be the family of intervals obtained by considering each edge of G as an interval of the line, closed between both its endpoints. A pair of $\{<\}$ -comparable edges in $E(G)$ correspond to a pair of disjoint intervals in $\mathcal{I}(G)$, a pair of $\{\square\}$ -comparable edges correspond to a pair of nesting intervals, and a pair of $\{\emptyset\}$ -comparable edges correspond to a pair of overlapping intervals. Note that this correspondence is bi-directional only if G is edge-disjoint, since a pair of edges sharing a vertex can correspond to a pair of nesting or overlapping intervals. Nevertheless, we can always modify $\mathcal{I}(G)$ in such a way that all intervals have unique endpoints, and so that any pair of

intervals who were sharing an endpoint now become non-nesting (resp. non-overlapping). A maximum pairwise disjoint subset of intervals can be computed in linear time using standard dynamic-programming, assuming the interval family is given in a sorted manner [21] (which we can provide in linear time in our case using bucket sorting). A maximum pairwise nesting subset can be computed in $O(m \log \log m)$ time in an interval family of m intervals (see for example the algorithm in [36]), and a maximum pairwise overlapping subset in $O(m^{1.5})$ time [34].

Lemma 2 ([21,36,34]). *Let G be a linear graph of size m . Then there exists*

- (1) *an $O(m)$ time algorithm for finding the largest $\{<\}$ -comparable subgraph of G ,*
- (2) *an $O(m \log \log m)$ time algorithm for finding the largest $\{\square\}$ -comparable subgraph of G , and*
- (3) *an $O(m^{1.5})$ time algorithm for finding the largest $\{\diamond\}$ -comparable subgraph of G .*

For any $R \in \{<, \square, \diamond\}$, finding the largest R -comparable subgraph in each input linear graph and returning the smallest found yields the following.

Corollary 3. *The MCSP problem for $\{<\}$ -structured patterns (resp. $\{\square\}$ -structured patterns and $\{\diamond\}$ -structured patterns) is solvable in $O(nm)$ (resp. $O(nm \log \log m)$ and $O(nm^{1.5})$) time, where $n = |\mathcal{G}|$ and $m = \max_{G \in \mathcal{G}} |E(G)|$.*

We next consider $\{<, \square\}$ -structured patterns. The MCSP problem for this type of patterns was considered in [11,28], in the context of multiple RNA structural alignment.

Theorem 4 ([28]). *The MCSP problem for $\{<, \square\}$ -structured patterns is **NP-hard** even if each input linear graph is a sequence of towers of height at most 2.*

Note, however, that the MCSP problem is polynomial-time solvable in case the number of input linear graphs is a constant [28]. The MCSP problem for $\{<, \square\}$ -structured patterns was proved to be approximable with ratio $O(\log^2 k)$ [11], where k is the size of the optimal solution. The approximation ratio was later improved to $\log k + 1$ [28].

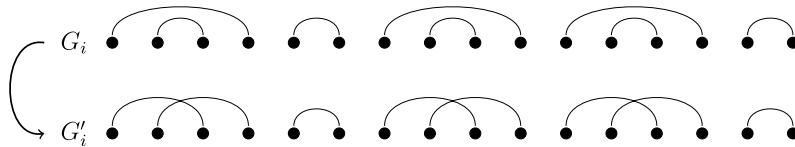
Theorem 5 ([28]). *The MCSP problem for $\{<, \square\}$ -structured patterns is approximable within ratio $O(\log k)$ in $O(nm^2)$ time, where k is the size of an optimal solution, $n = |\mathcal{G}|$, and m is the maximum size of any graph in \mathcal{G} .*

3. $\{<, \diamond\}$ -structured patterns

We now turn to considering the MCSP problem for $\{<, \diamond\}$ -structured patterns. We begin by proving a tight hardness result for the problem. Following this, we present an approximation algorithm for the problem which achieves an approximation ratio of $2\mathcal{H}(k)$ in $O(nm^{3.5} \log m)$ time, where k is the size of an optimal solution, $\mathcal{H}(k) = \sum_{i=1}^k 1/i$, $n = |\mathcal{G}|$, and m is the maximum size of any graph in \mathcal{G} .

Theorem 6. *The MCSP problem for $\{<, \diamond\}$ -structured patterns is **NP-hard** even if each input linear graph is a sequence of staircases of depth at most 2.*

Proof. We reduce from the MCSP problem for $\{<, \square\}$ -structured patterns, which is **NP-hard** even if each input graph is a sequence of towers of height at most 2 (Theorem 4). Let an arbitrary instance of MCSP for $\{<, \square\}$ -structured patterns be given by a family $\mathcal{G} = \{G_1, G_2, \dots, G_n\}$ of n sequences of towers with height at most 2. The corresponding instance of the MCSP problem for $\{<, \diamond\}$ -structured patterns consists of a family $\mathcal{G}' = \{G'_1, G'_2, \dots, G'_n\}$ of n sequences of staircases of depth at most 2, where for each i , $1 \leq i \leq n$, the graph G'_i is obtained from G_i by transforming each pair of $\{\square\}$ -comparable edges into a pair of $\{\diamond\}$ -comparable edges. Below is an example of such a transformation.



Clearly, our construction can be carried out in polynomial-time. Furthermore, there exists a common $\{<, \square\}$ -structured pattern of size k in \mathcal{G} if and only if there exists a common $\{<, \diamond\}$ -structured pattern of size k in \mathcal{G}' . \square

A recent result [29] implies that the MCSP problem for $\{<, \diamond\}$ -structured patterns is hard even if \mathcal{G} consists of only two linear graphs. However, the input linear graphs used in [29] are of unlimited structure, unlike in the lemma above. The case $|\mathcal{G}| = 1$ has been recently proved to be **NP-hard** [29].

We now show that the MCSP problem for $\{<, \diamond\}$ -structured is approximable within ratio $2\mathcal{H}(k)$. The first ingredient of our proof is to observe that any $\{<, \diamond\}$ -structured pattern contains a sequence of staircases of substantial size.

Lemma 7. *Let G be a $\{<, \diamond\}$ -comparable linear graph. There exists a partition $E(G) = E_{RED} \cup E_{BLUE}$ such that both $G[E_{RED}]$ and $G[E_{BLUE}]$ are sequences of staircases.*

Proof. The proof is constructive and uses the following edge-coloring algorithm: first, sort the edges of G by ascending left vertex. Let $E(G) = (e_1, e_2, \dots, e_m)$ be the edges of G according to that order. Next, start coloring the edges one at a time, proceeding from e_1 to e_m , in the following manner: start by initializing e_1 to be the *current leading edge*, and RED to be the

current color. Color the current leading edge with the current color, and proceed to color all following edges with the current color, as long as they are crossing with the current leading edge. As soon as an edge cannot be colored with the current color (this edge and the current leading edge are not crossing), change the current color (RED to BLUE or BLUE to RED), set this edge as the current leading edge, and continue in a similar fashion.

Let us say that two edges e_i and e_j are colored during the same run if there does not exist a leading edge e_ℓ with $i < \ell < j$. The following facts are immediate consequences of the algorithm: (i) each edge is colored either RED or BLUE, (ii) any two edges in the same run are colored by the same color and are $\{\emptyset\}$ -comparable, and (iii) any two edges colored in different runs with the same color are $\{<\}$ -comparable. Let E_{RED} and E_{BLUE} be the set of edges colored RED and BLUE respectively. By the above, it follows that $E_{\text{RED}} \cup E_{\text{BLUE}}$ is a partition of $E(G)$, and $G[E_{\text{RED}}]$ and $G[E_{\text{BLUE}}]$ are both sequences of staircases. \square

The second ingredient of our approach consists in showing that any sequence of staircases contains a balanced subgraph of substantial size.

Lemma 8. *Let G be a sequence of staircases of size k . Then G contains a balanced sequence of staircases with at least $\frac{k}{\mathcal{H}(k)}$ edges.*

Proof. Let G_1, G_2, \dots, G_p be the (maximal) staircases of G . For each $1 \leq i \leq k$, define $n_i = |\{G_j : 1 \leq j \leq p \wedge |E(G_j)| \geq i\}|$. According to this notation, $k = \sum_{i=1}^k n_i$ since the contribution of each staircase $G_j, 1 \leq j \leq p$, to the sum $\sum_{i=1}^k n_i$ is distributed over all n_i 's such that $1 \leq i \leq |E(G_j)|$.

Let k' be the size of the largest balanced sequence of staircases that occurs in G . We now make the crucial observation that G does contain a balanced sequence of staircases of size $i n_i$ for each $1 \leq i \leq k$ (by definition of the n_i 's). Moreover, no balanced sequence of staircases of G can have more than k' edges since k' is the size of an optimal solution. Then it follows that $i n_i \leq k'$ for each $1 \leq i \leq k$. Therefore, $k = \sum_{i=1}^k n_i \leq \sum_{i=1}^k \frac{k'}{i} = k' \sum_{i=1}^k \frac{1}{i} = k' \mathcal{H}(k)$, and the lemma is proved. \square

Note that $\mathcal{H}(k)$ is bounded by $\ln k + O(1)$. Combining Lemmas 7 and 8, we obtain the following.

Corollary 9. *Any $\{<, \emptyset\}$ -comparable linear graph of size k contains as a subgraph a balanced sequence of staircases of size at least $\frac{k}{2 \cdot \mathcal{H}(k)}$.*

What is left is to show that, given a set of linear graphs, one can find in polynomial-time the size of the largest balanced sequence of staircases that occurs in each input linear graph. Devoted to this particular purpose is Algorithm Bal-Seq-Staircase for finding a balanced sequence of staircases of width w and depth d in a linear graph G . For a linear graph $G \in \mathcal{G}$, and two integers i and j with $1 \leq i < j \leq |V(G)|$, we use $G[i, \dots, j]$ to denote the subgraph of G obtained by deleting all vertices labeled k with $k < i$ or $j < k$.

Bal-Seq-Staircase(G, d, w)

Input: A linear graph G of size m , and two positive integers d and w .

Result: true if and only if G contains a balanced sequence of staircases of width w and depth d .

- (1) $E' \leftarrow \emptyset$.
- (2) for $i = 1, 2, \dots, m - 1$ do
 - (a) Let j be the smallest integer such that $G[i \dots j]$ contains as a subgraph a staircase of size d (set $j = \infty$ if no such integer exists).
 - (b) if $j \neq \infty$ then $E' \leftarrow E' \cup \{(i, j)\}$.
- (3) Compute H , the maximum $\{<\}$ -comparable subgraph of the linear graph $G' = (V(G), E')$.
- (4) if $|E(H)| \geq w$ then return true else return false.

Lemma 10. *Algorithm Bal-Seq-Staircase(G, w, d) runs in $O(m^{2.5} \log m)$ time and returns true if and only if G contains a balanced sequence of staircases of width w and depth d .*

Proof. We first prove correctness of Algorithm Bal-Seq-Staircase. Clearly, G' is a linear graph. We now observe that each edge $(i, j) \in E'$ denotes the presence in G of a staircase of size d starting at vertex i' , $i \leq i' < j$, and ending at vertex j . Then it follows that, for any two $\{<\}$ -comparable edges (i, j) and (i', j') , $j < i'$, of E' , no edge of G can occur both in a staircase of size d denoted by edge (i, j) in E' and in a staircase (of size d) denoted by edge (i', j') in E' . Therefore, according to Step 4, Algorithm Bal-Seq-Staircase returns true if and only if G contains a balanced sequence of staircases of width w and depth d .

We now turn to evaluating the time complexity. Steps 2 and 3 are the only non-trivial steps of the algorithm. Checking whether subgraph $G[i \dots j]$ contains a staircase of size d can be done in $O(m^{1.5})$ time (Lemma 2). Hence, using a simple binary search strategy, one can find for any i the smallest j such that $G[i \dots j]$ contains a staircase of size d in $O(m^{1.5} \log m)$ time. It follows that Step 2 in total requires $O(m^{2.5} \log m)$ time. Step 3 can be computed in $O(m)$ time (Lemma 2), the whole algorithm requires $O(m^{2.5} \log m)$ time. \square

For a set \mathcal{G} of n linear graphs, each of size at most m , one can find the largest balanced sequence of staircases that occurs in each linear graph in \mathcal{G} with $O(nm^2)$ calls to Algorithm Bal-Seq-Staircase. We show how to reduce to $O(nm)$ calls. The following straightforward property is crucial to this aim.

Property 11. *Let G be a linear graph. For a given w (resp. w'), let d (resp. d') be the largest integer such that G contains a balanced sequence of staircases of width w (resp. w') and depth d (resp. d'). If $w' \geq w$ then $d' \leq d$.*

Lemma 12. Let \mathcal{G} be a set of n linear graphs, each of size at most m . The largest balanced sequence of staircases that occurs in each $G \in \mathcal{G}$ can be found in $O(nm^{3.5} \log m)$ time.

Proof. The algorithm is as follows. Let d_1 be the largest integer such that each linear graph in \mathcal{G} contains a staircase of depth d_1 (a staircase is by definition a balanced sequence of staircases of width 1). The integer d_1 can be certainly computed using at most nm calls to Algorithm Bal-Seq-Staircase (for practical considerations, a simple binary search would actually reduce this first step to $O(n \log m)$ calls). Next, for w ranging from 2 to m , let d_w be the largest integer such that each linear graph in \mathcal{G} contains a balanced sequence of staircases of width w and depth d_w . Finally, the algorithm returns as a result $\max\{w d_w : 1 \leq w \leq m\}$.

Now observe that, according to Property 11, $d_{w+1} \leq d_w$ for $1 \leq w < m$, i.e., the integers d_w 's, $1 \leq w \leq m$, form a non-increasing sequence of integers. Therefore, assuming d_w , $1 \leq w < m$, has just been computed, d_{w+1} can be computed as follows:

- (1) Initialize d_{w+1} to d_w .
- (2) While $d_{w+1} > 0$ and \mathcal{G} does not contain a balanced sequence of staircases of width $w + 1$ and depth d_{w+1} , decrease d_{w+1} by one.

Since one can check whether \mathcal{G} contains a balanced sequence of staircases of width $w + 1$ and depth d_{w+1} with n calls to Algorithm Bal-Seq-Staircase, our algorithm, as a whole, uses $O(nm)$ calls to Algorithm Bal-Seq-Staircase. The time complexity now follows from Lemma 10. \square

Theorem 13. The MCSP problem for $\{<, \delta\}$ -structured patterns is approximable within ratio $2\mathcal{H}(k)$ in $O(nm^{3.5} \log m)$ time, where k is the size of an optimal solution, $n = |\mathcal{G}|$, and m is the maximum size of any linear graph in \mathcal{G} .

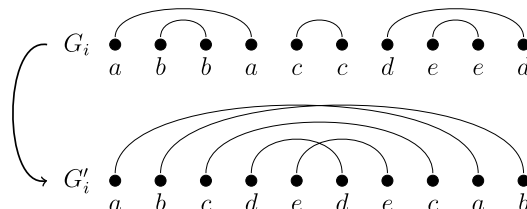
Proof. Corollary 9 and Lemma 12. \square

4. $\{\square, \delta\}$ -structured patterns

We now consider $\{\square, \delta\}$ -structured patterns. We begin by proving a hardness result, analogous to Theorem 6, which states that the MCSP problem for $\{\square, \delta\}$ -structured patterns is NP-hard even if the input consists of towers of staircases of depth at most 2. However, contrarily to the general approach we used for $\{<, \delta\}$ -structured patterns, we cannot use towers of staircases to obtain a good approximation ratio. Indeed, we show that there exists a $\{\square, \delta\}$ -comparable linear graph of size k which does not contain a tower of staircases of size $\varepsilon\sqrt{k}$ for some constant ε . On the other hand, such a graph must contain either a tower or a staircase with at least \sqrt{k} edges.

Theorem 14. The MCSP problem for $\{\square, \delta\}$ -structured patterns is NP-hard even if each input linear graph is a tower of staircases of depth at most 2.

Proof. As in proof of Theorem 6, we reduce from the MCSP problem for $\{<, \square\}$ -structured patterns which is NP-hard even if the input consists only of sequences of staircases, each of height at most 2 (Theorem 4). Let an arbitrary instance of the MCSP problem for $\{<, \square\}$ -structured patterns be given by a set $\mathcal{G} = \{G_1, G_2, \dots, G_n\}$ of n sequences of towers with height at most 2. The corresponding instance of the MCSP problem for $\{<, \delta\}$ -structured patterns consists of a set $\mathcal{G}' = \{G'_1, G'_2, \dots, G'_n\}$ of n towers of staircases of depth at most 2, where for each $i = 1, 2, \dots, n$, the graph G'_i is obtained from G_i by taking all the vertices which are right endpoints, reversing the order amongst them, and then placing them to the left of all left endpoints. Below is an example of such a transformation (labels are added for the sake of clarity).



It is easily seen that any linear graph G'_i is a tower of staircases of depth at most 2. Indeed, each tower is transformed into a staircase. Furthermore, any pair of edges in G_i are $\{<\}$ -comparable (resp. $\{\square\}$ -comparable) if and only if their corresponding edges in G'_i are $\{\square\}$ -comparable (resp. $\{\delta\}$ -comparable). Then it follows that there exists a common $\{<, \square\}$ -structured pattern of size k in \mathcal{G} if and only if there exists a common $\{\square, \delta\}$ -structured pattern of size k in \mathcal{G}' . \square

Note that the same theorem applies for staircases of towers, since if G is a tower of staircases then G' (as defined in the proof above) is a staircase of towers.

We next consider approximating the MCSP problem for $\{\square, \delta\}$ -structured patterns. First, let us observe the one-to-one correspondence between $\{\square, \delta\}$ -structured patterns and permutations. Let G be a $\{\square, \delta\}$ -comparable linear graph of size k . Then the vertices in G which are left endpoints of edges are labeled $\{1, \dots, k\}$ and the right endpoints are labeled $\{k + 1, \dots, 2k\}$. The permutation π_G corresponding to G is defined by $\pi_G(j - k) = i$ if and only if $(i, j) \in E(G)$.

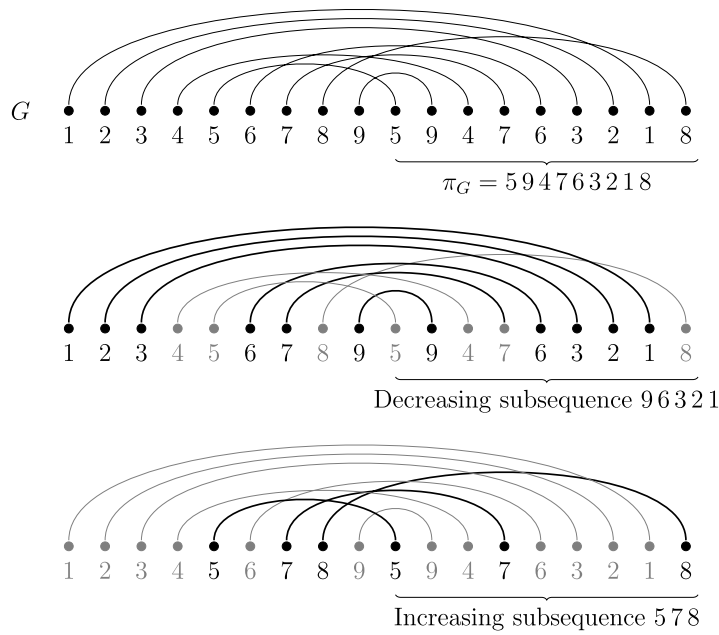


Fig. 3. A $\{\sqsubset, \emptyset\}$ -structured pattern G and the corresponding permutation $\pi_G = 594763218$. Also illustrated is the bijective correspondence between decreasing subsequences (resp. increasing subsequences) of π_G and $\{\sqsubset\}$ -structured (resp. $\{\emptyset\}$ -structured) patterns of G .

Clearly, all $\{\sqsubset, \emptyset\}$ -comparable linear graphs have corresponding permutations, and vice versa. It follows from this bijective correspondence that the number of different $\{\sqsubset, \emptyset\}$ -comparable linear graphs of size k is exactly $k!$. Moreover, notice that increasing subsequences in π_G correspond to $\{\emptyset\}$ -comparable subgraphs of G , while decreasing subsequences correspond to $\{\sqsubset\}$ -comparable subgraphs (see Fig. 3). More generally, a permutation $\sigma \in S_k$ is said to be a *pattern* (or to *occur*) within a permutation $\pi \in S_n$ if π has a subsequence that is order-isomorphic to σ [6], i.e., there exist indices $1 \leq i_1 \leq i_2 \leq \dots \leq i_k \leq n$ such that, for $1 \leq x \leq y \leq k$, $\pi_{i_x} < \pi_{i_y}$ if and only if $\sigma_x < \sigma_y$. If π does not contain σ , we say that π *avoids* σ , or that it is σ -avoiding. For example, $\pi = 24531$ contains 132 because the subsequence $\pi_1 \pi_3 \pi_4 = 253$ has the same relative order as 132 . However, $\pi = 42351$ is 132 -avoiding. In the light of the above bijection it is now clear that a linear graph H does not occur in another linear graph G if and only if π_G avoids π_H .

The well-known Erdős-Szekeres Theorem [16] states that any permutation on $1, \dots, k$ contains either an increasing or a decreasing subsequence of size at least \sqrt{k} . It is worth noticing that extremal Erdős-Szekeres (EES) permutations, i.e., permutations that do not contain monotone subsequences longer than \sqrt{k} , are known to exist (for example, there are 4 EES permutations of length 4: $2143, 2413, 3142$ and 3412). Hence, using the algorithms in Lemma 2 for finding the maximum common $\{\sqsubset\}$ -structured and $\{\emptyset\}$ -structured patterns, we obtain the following theorem.

Theorem 15. *The MCSP problem for model $\mathcal{M} = \{\sqsubset, \emptyset\}$ is approximable within ratio \sqrt{k} in $O(nm^{1.5})$ time, where k is the size of an optimal solution $n = |\mathcal{G}|$, and $m = \max_{G \in \mathcal{G}} |E(G)|$.*

For $k \in \mathbb{N}$ and $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$, let \mathcal{G}_k be a set of \mathcal{R} -comparable linear graphs of size k . Notice that the approximation algorithms we have offered and analyzed within this paper are all based on the key idea of identifying a family \mathcal{G}_k , $k \in \mathbb{N}$, of \mathcal{R} -comparable linear graphs with $|\mathcal{G}_k| \leq \text{poly}(k)$ and such that every \mathcal{R} -comparable linear graph of size at least $f(k)$ contains at least one member of \mathcal{G}_k . Clearly, this leads to a poly-time $\frac{k}{f(k)}$ -approximation algorithm whenever the function $\text{poly}(\cdot)$ is polynomial. We now want to first address the inherent limitations of this approach.

For $k \in \mathbb{N}$, let $\Pi_k \subseteq S_k$ be a set of $|\Pi_k|$ permutations on k elements. From what said above, each permutation in Π_k can be equivalently regarded as a $\{\sqsubset, \emptyset\}$ -comparable linear graph. Alon [1] recently communicated us a proof of essentially the following lemma.

Lemma 16 ([1]). *For every family of permutations $\Pi_k \subseteq S_k$, $k \in \mathbb{N}$ and $|\Pi_k| \leq 2^k$, there exists a permutation $\pi \in S_K$, $K = \Omega(k^2)$, which avoids all permutations in Π_k .*

Proof. The probability that a random (uniform) permutation in S_k is one of the permutations in Π_k is at most $2^k/k!$. Consider a random (uniform) permutation $\tilde{\pi}_K \in S_K$ and notice that there are at most $\binom{K}{k}$ distinct permutations in S_k which are subpermutations of $\tilde{\pi}_K$. Moreover, each of these subpermutations of $\tilde{\pi}_K$ is a random (uniform) permutation in S_k , though these subpermutations are not mutually independent.

Therefore, by the linearity of expectation (which does not require independence to hold), the expected number of permutations in Π_k which are subpermutations of $\tilde{\pi}_K$ is bounded by $\binom{K}{k} 2^k/k!$. Using Stirling’s formula to estimate this

expression, we get

$$\begin{aligned} \binom{K}{k} \frac{2^k}{k!} &= \frac{2^k K!}{(k!)^2 (K-k)!} \\ &\approx 2^k \left(\sqrt{2\pi K} \frac{K^K}{e^K} \right) \left(\frac{1}{\sqrt{2\pi k}} \frac{e^k}{k^k} \right)^2 \left(\frac{1}{\sqrt{2\pi (K-k)}} \frac{e^{K-k}}{(K-k)^{K-k}} \right) \\ &= \frac{1}{2\pi} \frac{e^k 2^k K^K}{k^{2k} (K-k)^{K-k}} \sqrt{\frac{K}{k^2 (K-k)}} \\ &= \frac{1}{2\pi} \left(\frac{2eK}{k^2} \right)^k \left(1 + \frac{k}{K-k} \right)^{\frac{K-k}{k} \cdot k} \sqrt{\frac{K}{k^2 (K-k)}} \\ &\approx \frac{1}{2\pi} \left(\frac{2e^2 K}{k^2} \right)^k \sqrt{\frac{K}{k^2 (K-k)}}. \end{aligned}$$

It is now easy to see that if $K = \varepsilon k^2$, for some positive constant $\varepsilon < (2e^2)^{-1}$, then $\binom{K}{k} \frac{2^k}{k!} < 1$. It follows that the expected number of permutations in Π_k which are subpermutations of a random permutation $\tilde{\pi}_K$ on $K = \varepsilon k^2$ elements is strictly less than 1. Thus there must exist a permutation $\pi_K \in S_K$ which avoids all permutations in Π_k , and so the lemma is proved. \square

Note that the above lemma shows the tightness of the positive approximability result offered in this section, at least within the general approach pursued within this paper. In particular, it shows that there exists a $\{<, \square\}$ -comparable linear graph of size $K = \Omega(k^2)$ which does not contain any $\{\square, \square\}$ -comparable linear graph out of a family of at most 2^k such graphs. Hence, even using more involved or interesting families of $\{\square, \square\}$ -comparable linear graphs to be used to probe our input graphs, no approximation guarantee better than $O(\sqrt{k})$ for maximum common $\{\square, \square\}$ -structured patterns can be possibly achieved.

5. General structured patterns

In this section we consider the MCSP problem for general patterns, i.e., $\{<, \square, \square\}$ -structured patterns. First, since $\{<, \square, \square\}$ -structured patterns generalize all other types of patterns, all hardness results presented in previous sections apply for general structured patterns as well. We present three approximation algorithms with increasing time complexities but decreasing approximation ratios. The first one achieves an approximation ratio of $O(k^{2/3})$ in $O(nm^{1.5})$ time while the second one achieves an approximation ratio of $O(\sqrt{k \log^2 k})$ in $O(nm^2)$ time. The third one is an $O(\sqrt{k \log k})$ -approximation algorithm which runs in $O(nm^{3.5} \log m)$ time. All algorithms rely on sufficiently large sub-patterns that occur in any $\{<, \square, \square\}$ -structured pattern, and the fact that finding maximum common structured patterns of these types is polynomial-time solvable.

We first observe that both relations $<$ and \square induce partial orders on the edges of a given linear graph. Recall now that a *chain* (resp. *anti-chain*) in a partial order is a subset of pairwise comparable (resp. incomparable) elements. Dilworth’s Theorem [13], which is a generalization of the Erdős–Szekeres Theorem [16], states that in any partial order, the size of the maximum chain equals the size of the minimum anti-chain partitioning. Therefore, in any partial order on k elements, the size of the maximum chain multiplied by the size of the maximum anti-chain is at least k . The following lemma states this property in our terms.

Lemma 17. *Let H be a $\{<, \square, \square\}$ -comparable linear graph of size k , width $w(H)$, and height $h(H)$. Also, let $hd(H)$ and $wd(H)$ be the sizes of the maximum $\{\square, \square\}$ -comparable and $\{<, \square\}$ -comparable subsets of $E(H)$. Then $k \leq w(H) \cdot hd(H)$ and $k \leq h(H) \cdot wd(H)$.*

An immediate consequence of Lemma 17 is as follows.

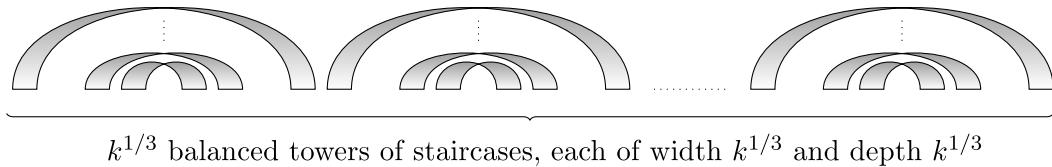
Lemma 18. *Let H be a $\{<, \square, \square\}$ -comparable linear graph of size k . Then H contains a simple structured pattern of size at least $k^{1/3}$.*

Proof. If the width of H is at least $k^{1/3}$ we are done. Otherwise, according to Lemma 17, H has a subgraph H' with at least $k^{2/3}$ edges which is $\{\square, \square\}$ -comparable. Applying Lemma 17 on H' , we obtain that H' has a subgraph with at least $(k^{2/3})^{1/2} = k^{1/3}$ edges which is either $\{\square\}$ -comparable or $\{\square\}$ -comparable. \square

Combining the above lemma with the fact that a maximum common simple structured pattern of \mathcal{G} can be found in $O(nm^{1.5})$ time (Corollary 3), we obtain our first approximation algorithm for general structured patterns.

Theorem 19. *The MCSP problem for $\{<, \square, \square\}$ -structured patterns is approximable within ratio $O(k^{2/3})$ in $O(nm^{1.5})$ time, where k is the size of an optimal solution, $n = |\mathcal{G}|$, and $m = \max_{G \in \mathcal{G}} |E(G)|$.*

It is easily seen, however, that Lemma 18 is tight. One way to obtain an extremal example of this is as follows: take $k^{1/3}$ balanced towers of staircases, each one of depth $k^{1/3}$ and height $k^{1/3}$, and concatenate them one next to the other into one supergraph of size k , reassigning labels accordingly.



Lemma 20. Let k be an integer such that $k^{1/3}$ is also integer. Then there exists a $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k that does not contain a simple structured pattern of size $\varepsilon k^{1/3}$ for any $\varepsilon > 1$.

Dilworth’s Theorem does not apply on the crossing relation since it is not transitive. However, an analogous result proven in [26] (see also [27]) implies that for any $\{<, \sqsubset, \emptyset\}$ -comparable linear graph H , $|E(H)| = O(d \cdot wh \log wh)$, where d and wh are sizes of the maximum $\{\emptyset\}$ -comparable and $\{<, \sqsubset\}$ -comparable subsets of $E(H)$, respectively. This yields the following analogue of Lemma 17.

Lemma 21. Let H be a $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k . Then H contains a subgraph of size $\Omega(\sqrt{k/\log k})$ which is either $\{<, \sqsubset\}$ -comparable or $\{\emptyset\}$ -comparable.

Using Lemma 21, the algorithm for finding a maximum $\{\emptyset\}$ -structured pattern given in Corollary 3, and the $O(\log k)$ -approximation algorithm for $\{<, \sqsubset\}$ -structured patterns given in Theorem 5, we obtain our second approximation algorithm.

Theorem 22. The MCSP problem for $\{<, \sqsubset, \emptyset\}$ -structured patterns is approximable within ratio $O(\sqrt{k \log^2 k})$ in $O(nm^2)$ time, where k is the size of an optimal solution, $n = |\mathcal{G}|$, and $m = \max_{G \in \mathcal{G}} |E(G)|$.

For our third algorithm, we show that any $\{<, \sqsubset, \emptyset\}$ -comparable linear graph contains a subgraph of sufficient size that is either a tower or a balanced sequence of staircases.

Lemma 23. Let H be a $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k . Then H contains either a tower or a balanced sequence of staircases of size $\Omega(\sqrt{k/\log k})$.

Proof. Let k_1 and k_2 denote the sizes of the maximum $\{\sqsubset\}$ -comparable and $\{<, \emptyset\}$ -comparable subgraphs of H , respectively.

According to Lemma 17, we have $k \leq k_1 k_2$. If $k_1 = \Omega(\sqrt{k/\log k})$, we are done. Otherwise, $k_1 = o(\sqrt{k/\log k})$ and so $k_2 = \Omega(\sqrt{k \log k})$. From Corollary 9 it follows that H contains a balanced sequence of staircases of size

$$\frac{k_2}{\mathcal{H}(k_2)} = \Omega\left(\frac{\sqrt{k \log k}}{\log \sqrt{k \log k}}\right) = \Omega(\sqrt{k/\log k}),$$

and the lemma follows. \square

Applying Lemma 23 and the algorithms for finding the maximum common tower and balanced sequence of staircases in \mathcal{G} given in Corollary 3 and Theorem 13, respectively, we obtain the following theorem.

Theorem 24. The MCSP problem for $\{<, \sqsubset, \emptyset\}$ -structured patterns is approximable within ratio $O(\sqrt{k \log k})$ in $O(nm^{3.5} \log m)$ time, where k is the size of an optimal solution, $n = |\mathcal{G}|$, and $m = \max_{G \in \mathcal{G}} |E(G)|$.

We next consider subgraphs of $\{<, \sqsubset, \emptyset\}$ -comparable linear graphs that are comparable by pairs of relations, i.e., by $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$ with $|\mathcal{R}| = 2$. We show that any $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k contains such a subgraph of size at least $0.39 k^{2/3}$, and that this lower bound is relatively tight. Unfortunately, this result cannot be applied for approximation purposes (approximating the MCSP problem for $\{\sqsubset, \emptyset\}$ -patterns remains the bottleneck). Nevertheless, we present this result on account of independent interest.

Lemma 25. Let H be a $\{<, \sqsubset, \emptyset\}$ -comparable graph of size k . Then H has a subgraph of size $\varepsilon k^{2/3}$, where $\varepsilon = \frac{\sqrt{17}-1}{8} \approx 0.39$, which is either $\{<, \sqsubset\}$ -comparable, $\{<, \emptyset\}$ -comparable, or $\{\sqsubset, \emptyset\}$ -comparable.

Proof. For simplicity, assume that both $k^{1/3}$ and $k^{2/3}$ are integers. Since H is $\{<, \sqsubset, \emptyset\}$ -comparable, we may assume that $V(H) = \{1, 2, \dots, 2k\}$. For each $x = 1, 2, \dots, 2k - 1$, let $E_x \subseteq E(H)$ be the set of edges (i, j) with $i \leq x < j$. Note that for all $x = 1, 2, \dots, 2k - 1$, $H[E_x]$, the subgraph of H obtained by deleting all edges $E \setminus E_x$, is a $\{\sqsubset, \emptyset\}$ -comparable subgraph of H . Therefore, if $|E_x| > \varepsilon k^{2/3}$ for some $x = 1, 2, \dots, 2k - 1$, we are done.

Otherwise, for $y = 1, 2, \dots, k^{1/3}$, let p_y be the smallest integer in $\{1, 2, \dots, 2k\}$ such that there are at least $k^{2/3}$ edges $(i, j) \in E(H)$ with $p_{y-1} \leq i < p_y$, where $p_0 = 1$. Also, for each $y = 1, 2, \dots, k^{1/3}$, let H_y be the subgraph of H made of edges (i, j) with $p_{y-1} \leq i < j < p_y$. Since $|E_x| \leq \varepsilon k^{2/3}$ for all $x = 1, \dots, 2k - 1$, then H_y contains at least $k^{2/3} - \varepsilon k^{2/3} = (1 - \varepsilon)k^{2/3}$ edges. According to Lemma 17, for each $y = 1, 2, \dots, k^{1/3}$, H_y contains either (i) a $\{<, \emptyset\}$ -comparable subgraph H'_y or (ii) a $\{\sqsubset\}$ -comparable subgraph H''_y of size at least $((1 - \varepsilon)k^{2/3})^{1/2} = (1 - \varepsilon)^{1/2} k^{1/3}$. Let Y be the set of those y ’s for which case (i) occurs. Then $H' = \cup_{y \in Y} H'_y$ is a $\{<, \emptyset\}$ -comparable subgraph of H and $H'' = \cup_{y \notin Y} H''_y$ is a $\{<, \sqsubset\}$ -comparable subgraph of H . Furthermore, at least one of H' and H'' has at least $2^{-1}(1 - \varepsilon)^{1/2} k^{1/3} k^{1/3} = 2^{-1}(1 - \varepsilon)^{1/2} k^{2/3}$ edges. But $2^{-1}(1 - \varepsilon)^{1/2} = \varepsilon$ for $\varepsilon = \frac{\sqrt{17}-1}{8} \approx 0.39$, which proves the lemma. \square

We believe the bound of Lemma 25 to be not the best possible. However, combining Lemmas 17 and 20, we show that the above lemma is relatively tight.

Lemma 26. *Let k be an integer such that $k^{1/3}$ is an integer. Then there exists a $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k that contains neither a $\{<, \sqsubset\}$ -comparable subgraph, nor a $\{<, \emptyset\}$ -comparable subgraph, nor a $\{\sqsubset, \emptyset\}$ -comparable subgraph of size least $\varepsilon k^{2/3}$ for any $\varepsilon > 1$.*

Proof. Suppose, aiming at a contradiction, that the lemma is false. Then any $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k contains a subgraph of size at least $\varepsilon k^{2/3}$, $\varepsilon > 1$, which is either $\{<, \sqsubset\}$ -comparable, $\{<, \emptyset\}$ -comparable, or $\{\sqsubset, \emptyset\}$ -comparable. Therefore, by Lemma 17, any $\{<, \sqsubset, \emptyset\}$ -comparable linear graph of size k contains a subgraph with at least $\sqrt{\varepsilon} k^{2/3} = \varepsilon^{1/2} k^{1/3}$ edges which is either $\{<\}$ -comparable, $\{\sqsubset\}$ -comparable, or $\{\emptyset\}$ -comparable. According to Lemma 20, this is the desired contradiction. \square

6. Discussion and open problems

In this paper, we introduced the MCSP problem as a general framework for many structure-comparison and structure-prediction problems, that occur mainly in computational molecular biology. Our framework followed the approach in [35] by analyzing all types of \mathcal{R} -structured patterns, $\mathcal{R} \subseteq \{<, \sqsubset, \emptyset\}$. We gave tight hardness results for finding maximum common $\{<, \emptyset\}$ -structured patterns and maximum common $\{<, \emptyset\}$ -structured patterns. We also proved that the MCSP problem is approximable within ratios: (i) $2\mathcal{H}(k)$ for $\{<, \emptyset\}$ -structured patterns, (ii) $k^{1/2}$ for $\{\sqsubset, \emptyset\}$ -structured patterns, and (iii) $O(\sqrt{k \log k})$ for $\{<, \sqsubset, \emptyset\}$ -structured patterns.

There are many questions left open by our study. Below we list some of them. According to Lemma 25, we could substantially improve in terms of approximation ratio on all the algorithms suggested for general structured patterns, if we had a better approximation algorithm for $\{\sqsubset, \emptyset\}$ -structured patterns. Is there an approximation algorithm which achieves a better ratio than the simple \sqrt{k} algorithm? On the same note, can lower bounds on the approximation factor of the MCSP problem for $\{<, \sqsubset, \emptyset\}$ -structured patterns or $\{\sqsubset, \emptyset\}$ -structured patterns be proven? How about $\{<, \sqsubset\}$ -structured patterns or $\{<, \emptyset\}$ -structured patterns? Besides, the last decade has seen an increasing development in the theory of parameterized complexity [14]. Can the MCSP problem be tackled better by applying this theory? Last but not least, the MCSP problem for $\{<, \emptyset\}$ -structured patterns is still open in case $|\mathcal{G}_1| = 1$, i.e., the case where the input consists of one linear graph G and one wishes to find the largest $\{<, \emptyset\}$ -comparable subgraph of G (see [5,10,35]). Is there a polynomial-time algorithm for this problem?

Acknowledgement

The second author was supported by the Adams Fellowship of the Israel Academy of Sciences and Humanities.

References

- [1] N. Alon, Personal communication, 2006.
- [2] L. Alonso, R. Schott, On the tree inclusion problem, in: A.M. Borzyszkowski, S. Sokolowski (Eds.), Proc. 18th Mathematical Foundations of Computer Science, MFCS, Gdansk, Poland, in: Lecture Notes in Computer Science, vol. 711, 1993, pp. 211–221.
- [3] A. Apostolico, C. Guerra, The longest common subsequence problem revisited, *Algorithmica* 2 (1987) 315–336.
- [4] G. Blin, G. Fertin, S. Vialette, What makes the arc-preserving subsequence problem hard? *LNCSTrans. Comput. Syst. Biol.* 2 (2005) 1–36.
- [5] G. Blin, G. Fertin, S. Vialette, Extracting constrained 2-interval subsets in 2-interval sets, *Theoret. Comput. Sci.* 385 (1–3) (2007) 241–263.
- [6] M. Bóna, *Combinatorics of permutations*, in: Discrete Mathematics and its Applications, Chapman & Hall/CRC, 2004.
- [7] P. Bose, J.F. Buss, A. Lubiw, Pattern matching for permutations, *Inform. Process. Lett.* 65 (5) (1998) 277–283.
- [8] W.Y.C. Chen, E.Y.P. Deng, R.P. Stanley, R.R.X. Du, C.H.F. Yan, Crossings and nestings of matchings and partitions, *Trans. Amer. Math. Soc.* 359 (4) (2007) 1555–1575.
- [9] M.-J. Chung, More efficient algorithm for ordered tree inclusion, *J. Algorithms* 26 (2) (1998) 370–385.
- [10] M. Crochemore, D. Hermelin, G. Landau, S. Vialette, Approximating the 2-interval pattern problem, in: Gerth S. Brodal, S. Leonardi (Eds.), Proc. 13th European Symposium on Algorithms, ESA, Palma de Mallorca, Spain, in: Lecture Notes in Computer Science, 2005.
- [11] E. Davydov, S. Batzoglou, A computational model for rna multiple structural alignment, *Theoret. Comput. Sci.* 368 (3) (2006) 205–216.
- [12] E. Demaine, S. Mozes, B. Rossman, O. Weimann, An optimal decomposition algorithm for tree edit distance, in: L. Arge, C. Cachin, T. Jurdzinski, A. Tarlecki (Eds.), Proc. 34th International Colloquium on Automata, Languages and Programming, ICALP, Wroclaw, Poland, in: Lecture Notes in Computer Science, vol. 4596, 2007, pp. 146–157.
- [13] R.P. Dilworth, A decomposition theorem for partially ordered sets, *Ann. of Math.* 51 (1950) 161–166.
- [14] R. Downey, M. Fellows, *Parameterized Complexity*, Springer-Verlag, 1999.
- [15] D. Erdstein, Z. Galil, R. Giancarlo, G.F. Italiano, Sparse dynamic programming I: linear cost functions, *J. ACM* 39 (3) (1992) 519–545.
- [16] P. Erdős, G. Szekeres, A combinatorial problem in geometry, *Compos. Math.* 2 (1935) 463–470.
- [17] P. Evans, Finding common subsequences with arcs and pseudoknots, in: M. Crochemore, M. Paterson (Eds.), Proc. 10th Annual Symposium Combinatorial Pattern Matching, CPM, Warwick University, UK, in: Lecture Notes in Computer Science, vol. 1645, Springer, 1999, pp. 270–280.
- [18] D. Goldman, S. Istrail, C.H. Papadimitriou, Algorithmic aspects of protein structure similarity, in: Proc. 40th Annual Symposium of Foundations of Computer Science, FOCS, New York, NY, USA, IEEE Computer Society, 1999, pp. 512–522.
- [19] J. Gramm, A polynomial-time algorithm for the matching of crossing contact-map patterns, *IEEE/ACM Trans. Comput. Biol. Bioinform.* 1 (4) (2004) 171–180.
- [20] J. Gramm, J. Guo, R. Niedermeier, Pattern matching for arc-annotated sequences, in: M. Agrawal, A. Seth (Eds.), Proc. 22nd Foundations of Software Technology and Theoret. Comput. Sci., FSTTCS, Kanpur, India, in: Lecture Notes in Computer Science, 2002, pp. 182–193.
- [21] U.I. Gupta, D.T. Lee, J.Y.-T. Leung, Efficient algorithms for interval graph and circular-arc graphs, *Networks* 12 (1982) 459–467.
- [22] D.S. Hirschberg, Algorithms for the longest common subsequence problem, *J. ACM* 24 (4) (1977) 664–675.

- [23] J.W. Hunt, T.G. Szymanski, A fast algorithm for computing longest common subsequences, *Commun. ACM* 20 (1977) 350–353.
- [24] P. Kärpeläinen, H. Mannila, Ordered and unordered tree inclusion, *SIAM J. Comp.* 24 (2) (1995) 340–356.
- [25] P.N. Klein, Computing the edit-distance between unrooted ordered trees, in: G. Bilardi, G.F. Italiano, A. Pietracaprina, G. Pucci (Eds.), *Proc. 6th European Symposium on Algorithms, ESA, Venice, Italy*, in: *Lecture Notes in Computer Science*, vol. 1461, 1998, pp. 91–102.
- [26] A.V. Kostochka, On upper bounds for the chromatic numbers of graphs, *Tr. Inst. Mat.* 10 (1988) 204–226 (in Russian).
- [27] A. Kostochka, J. Kratochvíl, Covering and coloring polygon-circle graphs, *Discrete Math.* 163 (1997) 299–305.
- [28] M. Kubica, R. Rizzi, S. Vialette, T. Waleń, Approximation of RNA multiple structural alignment, in: M. Lewenstein, G. Valiente (Eds.), *Proc. 17th Annual Symposium on Combinatorial Pattern Matching, CPM, Barcelona, Spain*, in: *Lecture Notes in Computer Science*, vol. 4009, Springer, 2006.
- [29] S.C. Li, M. Li, On two open problems of 2-interval patterns, *Theoret. Comput. Sci.* 410 (24–25) (2009) 2410–2423.
- [30] D. Maier, The complexity of some problems on subsequences and supersequences, *J. ACM* 25 (2) (1978) 322–336.
- [31] W.J. Masek, M.S. Paterson, A faster algorithm computing string edit distances, *J. Comput. Syst. Sci.* 20 (1) (1980) 18–31.
- [32] D. Shasha, K. Zhang, Simple fast algorithms for the editing distance between trees and related problems, *SIAM J. Comput.* 18 (6) (1989) 1245–1262.
- [33] R.P. Stanley, *Enumerative Combinatorics*, second ed., vol. 2, Cambridge University Press, 1997.
- [34] A. Tiskin, Longest common subsequences in permutations and maximum cliques in circle graphs, in: M. Lewenstein, G. Valiente (Eds.), *Proc. 17th Combinatorial Pattern Matching, CPM, Barcelona, Spain*, in: *Lecture Notes in Computer Science*, vol. 4009, 2006, pp. 270–281.
- [35] S. Vialette, On the computational complexity of 2-interval pattern matching problems, *Theoret. Comput. Sci.* 312 (2–3) (2004) 223–249.
- [36] M.-S. Whang, G.-H. Wang, Efficient algorithms for the maximum weight clique and maximum weight independent set problems on permutation graphs, *Inform. Process. Lett.* 43 (1992) 293–295.