




# Coloring and Recognizing Mixed Interval Graphs

Grzegorz Gutowski   



Theoretical Computer Science Department, Faculty of Mathematics and Computer Science,  
Jagiellonian University, Kraków, Poland

Konstanty Junosza-Szaniawski  

Warsaw University of Technology, Poland

Felix Klesen  

Universität Würzburg, Germany



Paweł Rzażewski  

Warsaw University of Technology, Poland

Institute of Informatics, University of Warsaw, Poland

Alexander Wolff  

Universität Würzburg, Germany

Johannes Zink  

Universität Würzburg, Germany

---

## Abstract

A *mixed interval graph* is an interval graph that has, for every pair of intersecting intervals, either an arc (directed arbitrarily) or an (undirected) edge. We are particularly interested in scenarios where edges and arcs are defined by the geometry of intervals. In a proper coloring of a mixed interval graph  $G$ , an interval  $u$  receives a lower (different) color than an interval  $v$  if  $G$  contains arc  $(u, v)$  (edge  $\{u, v\}$ ). Coloring of mixed graphs has applications, for example, in scheduling with precedence constraints; see a survey by Sotskov [Mathematics, 2020].

For coloring general mixed interval graphs, we present a  $\min\{\omega(G), \lambda(G) + 1\}$ -approximation algorithm, where  $\omega(G)$  is the size of a largest clique and  $\lambda(G)$  is the length of a longest directed path in  $G$ . For the subclass of *bidirectional interval graphs* (introduced recently for an application in graph drawing), we show that optimal coloring is NP-hard. This was known for general mixed interval graphs.

We introduce a new natural class of mixed interval graphs, which we call *containment interval graphs*. In such a graph, there is an arc  $(u, v)$  if interval  $u$  contains interval  $v$ , and there is an edge  $\{u, v\}$  if  $u$  and  $v$  overlap. We show that these graphs can be recognized in polynomial time, that coloring them with the minimum number of colors is NP-hard, and that there is a 2-approximation algorithm for coloring.

**2012 ACM Subject Classification** Mathematics of computing  $\rightarrow$  Graph theory

**Keywords and phrases** Interval Graphs, Mixed Graphs, Graph Coloring

**Digital Object Identifier** 10.4230/LIPIcs.ISAAC.2023.36

**Related Version** *Full Version*: <https://arxiv.org/abs/2303.07960>

**Funding** *Grzegorz Gutowski*: partially supported by the National Science Center of Poland under grant no. 2019/35/B/ST6/02472.

*Johannes Zink*: partially supported by DFG grant Wo 758/11-1.

**Acknowledgements** We are indebted to Krzysztof Fleszar, Zbigniew Lonc, Karolina Okrasa, and Marta Piecyk for fruitful discussions. Additionally, we acknowledge the welcoming and productive atmosphere at the workshop Homonolo 2022, where some of the work was done.



© Grzegorz Gutowski, Konstanty Junosza-Szaniawski, Felix Klesen, Paweł Rzażewski, Alexander Wolff, and Johannes Zink;

licensed under Creative Commons License CC-BY 4.0

34th International Symposium on Algorithms and Computation (ISAAC 2023).

Editors: Satoru Iwata and Naonori Kakimura; Article No. 36; pp. 36:1–36:14



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

In a geometric intersection graph, the vertices represent geometric objects, and two vertices are adjacent if and only if the corresponding objects intersect. For example, *interval graphs* are the intersection graphs of intervals on the real line. These graphs are well understood: interval graphs are *chordal* and can thus be colored optimally (that is, with the least number of colors) in polynomial time. In other words, given an interval graph  $G$ , its *chromatic number*  $\chi(G)$  can be computed efficiently.

The notion of coloring can be adapted to directed graphs where an arc  $(u, v)$  means that the color of  $u$  must be smaller than that of  $v$ . Clearly, such a coloring can only exist if the given graph is acyclic. Given a directed acyclic graph, its chromatic number can be computed efficiently (via topological sorting).

A generalization of both undirected and directed graphs are *mixed graphs* that have edges and arcs. A *proper coloring* of a mixed graph  $G$  with vertex set  $V(G)$  is a function  $f: V(G) \rightarrow \mathbb{N}$  such that, for any distinct vertices  $u$  and  $v$  of  $G$ , the following conditions hold:

1. if there is an edge  $\{u, v\}$ , then  $f(u) \neq f(v)$ , and
2. if there is an arc  $(u, v)$ , then  $f(u) < f(v)$ .

The objective is to minimize the number of colors.

The concept of mixed graphs was introduced by Sotskov and Tanaev [16] and reintroduced by Hansen, Kuplinsky, and de Werra [9] in the context of proper colorings of mixed graphs. Properly coloring mixed graphs is NP-hard even for bipartite planar graphs [12] but admits efficient algorithms for trees [5] and series-parallel graphs [6].

For a mixed interval graph  $G$ , the *underlying undirected graph* of  $G$ , denoted by  $U(G)$ , has an edge for every edge or arc of  $G$ . Note that testing whether a given graph  $G$  is a mixed interval graph means testing whether  $U(G)$  is an interval graph, which takes linear time [10].

**Motivation.** Mixed graphs are graphs where some vertices are connected by (undirected) edges and others by directed arcs. Such structures are useful for modeling relationships that involve both directed and undirected connections and find applications in various areas, including network analysis, transportation planning, job scheduling, and circuit design.

Coloring mixed graphs is relevant in task scheduling problems where tasks have dependencies and resource requirements [18, 3, 15]. In circuit design, coloring mixed graphs allows us to reduce signal crosstalk or interference. Other applications include modeling of metabolic pathways in biology [1], process management in operating systems [2], traffic signal synchronization [13], and timetabling [4]. See the extensive survey by Sotskov [14] for other applications and relevant problems. Coloring of mixed graphs is a challenging problem, as many techniques known for solving graph coloring problems fail in the more general setting.

The study of mixed graph coloring for interval graphs was initiated by Zink et al. [19], motivated by the minimization of the number of additional sub-layers for routing edges in layered orthogonal graph drawing according to the so-called Sugiyama framework [17]. In a follow-up paper, Gutowski et al. [8] resolved some of the problems concerning interval graphs where the subset of arcs and their orientations are given by the geometry of the intersecting intervals. Driven by the graph drawing application, they focused on the *directional variant* where, for every pair of intersecting intervals, there is an edge when one interval is contained in the other and there is an arc oriented towards the right interval when the intervals overlap.

In this paper we focus on the *containment variant* where, for any pair of intersecting intervals, there is an arc oriented towards the smaller interval when one interval is contained in the other, and an edge when they overlap. This is the only other natural geometric variant

■ **Table 1** Known and new results concerning subclasses of mixed interval graphs. The time complexities refer to a given set of  $n$  intervals with  $m$  pairwise intersections. (We use T, P, and C as shorthand for Theorem, Proposition, Corollary, respectively.)

Mixed interval graph class	complexity	Coloring						Recognition		
		lower bound		upper bound		approximation				
containment	NP-hard	T7	$2\omega-1$	P6	$2\omega-1$	T2	2	C5	$O(nm)$	T1
directional	$O(n \log n)$	[8]					1	[8]	$O(n^2)$	[8]
bidirectional	NP-hard	T8					2	[8]	open	
general	NP-hard	[8]	$(\lambda+2)\omega/2$	P10	$(\lambda+1)\omega$	T9	$\min\{\omega, \lambda+1\}$	T9	$O(n+m)$	[10]

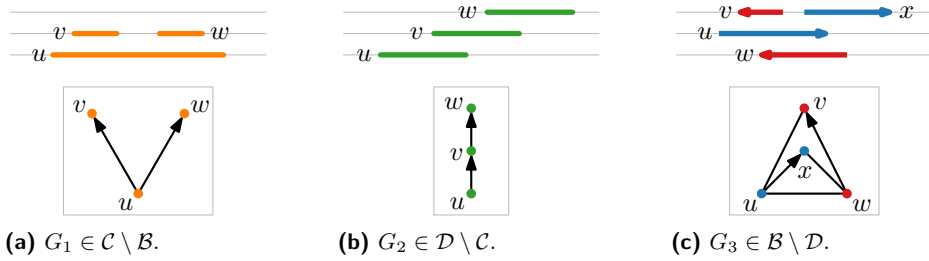
that can be defined for interval graphs, but the containment variant can also be defined for other geometric intersection graphs, or even for graphs defined by systems of intersecting sets.

As there are already some effective techniques for the directional variant, our hope was to use them in the containment variant, or even in more general settings. Quite unexpectedly, the containment variant for interval graphs turned out to be more difficult than the directional variant. We have found our techniques for proving lower bounds for the containment variant of interval graphs to be applicable for the *bidirectional variant* considered previously [19, 8]. This variant is a generalization of the directional variant mentioned above. Every interval has an orientation; left-going or right-going. There is an arc between two intervals if and only if they overlap and their orientations agree. Arcs between left-going intervals are directed as in the directional variant; the condition for right-going intervals is symmetric. As a result, we get that minimizing the number of additional sub-layers in layered orthogonal graph drawing according to the Sugiyama framework is NP-hard.

**Our Contribution.** In this paper we forward the study of coloring mixed graphs where edge directions have a geometric meaning. To this end, we introduce a new natural class of mixed interval graphs, which we call *containment interval graphs*. In such a graph, there is an arc  $(u, v)$  if interval  $u$  contains interval  $v$ , and there is an edge  $\{u, v\}$  if  $u$  and  $v$  overlap. For a set  $\mathcal{I}$  of intervals, let  $\mathcal{C}[\mathcal{I}]$  be the containment interval graph induced by  $\mathcal{I}$ . We show that these graphs can be recognized in polynomial time (Section 2), that coloring them optimally is NP-hard (Section 4), and that, for every set  $\mathcal{I}$  of intervals, it holds that  $\chi(\mathcal{C}[\mathcal{I}]) \leq 2\omega(\mathcal{C}[\mathcal{I}]) - 1$ , that is,  $\mathcal{C}[\mathcal{I}]$  can be colored with fewer than twice as many colors as the size of the largest clique in  $\mathcal{C}[\mathcal{I}]$  (Section 3). In other words, containment interval graphs are  $\chi$ -bounded. Our constructive proof yields a 2-approximation algorithm for coloring containment interval graphs.

Then we prove that, for the class of bidirectional interval graphs, optimal coloring is NP-hard (Section 5). This answers (negatively) an open problem that was asked previously [8]. Our reduction is similar to the one for containment interval graphs, but technically somewhat more challenging. Finally, we show that, for any mixed interval graph  $G$  without directed cycles, it holds that  $\chi(G) \leq \omega(G) \cdot (\lambda(G) + 1)$ , where  $\lambda(G)$  denotes the length of a longest directed path in  $G$  (Section 6). Since  $\chi(G) \geq \max\{\omega(G), \lambda(G) + 1\}$ , our constructive proof for the upper bound yields a  $\min\{\omega(G), \lambda(G) + 1\}$ -approximation algorithm. The upper bound is asymptotically tight in the worst case.

Table 1 gives an overview over known and new results concerning the above-mentioned subclasses of mixed interval graphs. Given a positive integer  $k$ , we use  $[k]$  as shorthand for the set  $\{1, 2, \dots, k\}$ . When we visualize a graph coloring corresponding to a set of intervals,



■ **Figure 1** Let  $\mathcal{D}$ ,  $\mathcal{B}$ , and  $\mathcal{C}$  be the classes of directional, bidirectional, and containment interval graphs. Clearly,  $\mathcal{D} \subseteq \mathcal{B}$ . The above sets of intervals and the corresponding directed graphs show that the classes  $\mathcal{D}$  and  $\mathcal{B}$  are incomparable with the class  $\mathcal{C}$  and that  $\mathcal{D}$  is properly contained in  $\mathcal{B}$ .

we use horizontal tracks to indicate the color. In Figure 1, we briefly analyze the relationships between the three classes of geometrically defined mixed interval graphs; directional ( $\mathcal{D}$ ), bidirectional ( $\mathcal{B}$ ), and containment interval graphs ( $\mathcal{C}$ ).

## 2 Recognition of Containment Interval Graphs

Booth and Lueker [10] introduced a data structure called *PQ-tree* to recognize, for a given undirected graph  $G$ , whether  $G$  is an interval graph. A PQ-tree is a rooted tree of so-called *P-nodes*, where the order of children can be arbitrarily permuted, and *Q-nodes*, where the order of children is fixed up to inversion. A specific permutation of all nodes is called a *rotation*. One can think of the leaves of a PQ-tree to represent the maximal cliques of  $G$  and a specific rotation to represent an order of the maximal cliques, which implies an interval representation of  $G$  where every vertex is contained in a consecutive sequence of maximal cliques. (Actually, a PQ-tree can encode all possible interval representations of  $G$ .)

Observe that a representation of a containment interval graph is an interval representation. Hence, if, for a given mixed graph  $G$ , a containment representation  $\mathcal{I}$  exists, then  $\mathcal{I}$  corresponds to a rotation of the PQ-tree constructed for the underlying undirected graph  $U(G)$  by the algorithm of Booth and Lueker [10]. Hence, to recognize a containment interval graph  $G$ , we proceed in three phases. First, we compute a PQ-tree  $T$  of  $U(G)$ . Second, we find a rotation of  $T$  corresponding to a containment representation of  $G$ . Third, we determine suitable endpoints for the intervals corresponding to our selected rotation resulting in a containment representation  $\mathcal{I}$  of  $G$ .

In the second phase, we proceed top-down to fix the permutation of each node of  $T$  while we maintain as invariant that before and after deciding the permutation of a single node, we can still reach a rotation of  $T$  corresponding to a containment representation  $\mathcal{I}$  (provided  $G$  is a containment interval graph). Depending on the set of maximal cliques (corresponding to leaves) a vertex  $v$  is contained in, we can determine where  $v$  is *introduced* in  $T$  (roughly at the root of the subtree containing all leaves corresponding to  $v$ ). Intuitively, it is “natural” for a vertex  $u$  introduced further up in the tree to have an arc towards a vertex  $v$  introduced further down in the tree. However, if  $u$  and  $v$  are connected by an edge, we need to permute the nodes of the PQ-tree such that both  $u$  and  $v$  start or end in the same maximal clique. These restrictions can propagate.

If we end up with a rotation of  $T$ , we construct, in the third phase, a corresponding containment representation if possible. To this end, we determine for every vertex the first and the last clique it appears in, which groups the left and right endpoint of the intervals. Within each group, we sort the endpoints according to the constraints implied by the arcs

and edges where possible. What remains are induced mixed subgraphs of vertices that start and end in the same cliques and that behave the same with respect to every other vertex (i.e., they are all connected to this vertex by an outgoing arc or an incoming arc or an edge). We can interpret each such subgraph as a *partially ordered set* for which we need to check whether it is *two-dimensional* and find two corresponding linear orders, which gives us an ordering of their left and their right endpoints. This last part depends on the linear-time algorithm by McConnell and Spinrad [11] that can construct such two orders for any two-dimensional poset.

► **Theorem 1.** *There is an algorithm that, given a mixed graph  $G$ , decides whether  $G$  is a containment interval graph. The algorithm runs in  $O(nm)$  time, where  $n$  is the number of vertices of  $G$  and  $m$  is the total number of edges and arcs of  $G$ , and produces a containment representation of  $G$  if  $G$  admits one.*

The full proof follows the ideas presented above, but has some technical subtleties; see the full version of this article [7].

### 3 A 2-Approximation Algorithm for Coloring Containment Interval Graphs

In this section, we present a 2-approximation algorithm for coloring containment interval graphs, we detail how to make the algorithm run in  $O(n \log n)$  time for a set of  $n$  intervals, and we construct a family of sets of intervals that shows that our analysis is tight.

► **Theorem 2.** *For any set  $\mathcal{I}$  of intervals, the containment interval graph  $\mathcal{C}[\mathcal{I}]$  induced by  $\mathcal{I}$  admits a proper coloring with at most  $2 \cdot \omega(\mathcal{C}[\mathcal{I}]) - 1$  colors.*

**Proof.** For simplicity, let  $G := \mathcal{C}[\mathcal{I}]$  and  $\omega := \omega(G)$ . We use induction on  $\omega$ . If  $\omega = 1$ , then  $G$  has no edges and clearly admits a proper coloring using only one color. So assume that  $\omega > 1$  and that the theorem holds for all graphs with smaller clique number.

Recall that a *proper* interval graph is an interval graph that has a representation where no interval is contained in another interval. Let  $M(\mathcal{I})$  denote the subset of  $\mathcal{I}$  consisting of intervals that are maximal with respect to the containment relation. In particular,  $\mathcal{C}[M(\mathcal{I})]$  is a proper interval graph. Observe that  $\bigcup M(\mathcal{I}) = \bigcup \mathcal{I}$  (where we consider the union of intervals as a subset of the real line). Let  $R$  be an inclusion-wise minimal subset of  $M(\mathcal{I})$  such that  $\bigcup R = \bigcup \mathcal{I}$ . In Figure 2, the intervals in  $M(\mathcal{I})$  are marked with crosses and the set of intervals on the lowest two (gray) lines is one way of choosing  $R$ .

▷ **Claim 3.**  $\mathcal{C}[R]$  is an undirected linear forest.

**Proof.** All intervals in  $M(\mathcal{I})$  and thus in  $R$  are incomparable with respect to the containment relation, so  $\mathcal{C}[R]$  has no arcs. Note that  $\mathcal{C}[R]$  is a proper interval graph, so it contains no induced  $K_{1,3}$  and no induced cycle with at least four vertices. Thus it suffices to prove that  $\mathcal{C}[R]$  is triangle-free. For contradiction, suppose otherwise. Let  $x, y, z$  induce a triangle in  $\mathcal{C}[R]$ , ordered according to their left endpoints. As  $x, y, z$  are pairwise overlapping, note that  $y \subseteq x \cup z$ , and thus  $\bigcup (R \setminus \{y\}) = \bigcup R$ . This contradicts the minimality of  $R$ . ◁

By the claim above,  $\mathcal{C}[R]$  can be properly colored with colors  $\{1, 2\}$ . Let  $f_1$  be such a coloring. If  $R = \mathcal{I}$ , we are done (using only  $\omega$  many colors), so suppose that  $\mathcal{I} \setminus R \neq \emptyset$ . Slightly abusing notation, we define  $G' := G - R$ .

▷ **Claim 4.** The largest clique in  $G'$  has at most  $\omega - 1$  vertices.



■ **Figure 2** A set of intervals and a coloring produced by the 2-approximation algorithm. The intervals that lie in  $M(\mathcal{I})$  at the top level of the recursion are marked with crosses.

Proof. As  $G'$  is a subgraph of  $G$ , each clique in  $G'$  has at most  $\omega$  vertices. For contradiction, suppose that there is a set  $S \subseteq \mathcal{I} \setminus R$  such that  $|S| = \omega$  and all intervals in  $S$  pairwise intersect. By the Helly property of intervals,  $\bigcap S \neq \emptyset$ . Let  $p \in \bigcap S$ . Since  $\bigcup R = \bigcup \mathcal{I}$ , there is an interval  $r \in R$  that contains  $p$ . Thus  $S \cup \{r\}$  is a clique in  $G$  with  $\omega + 1$  vertices, which contradicts the definition of  $\omega$ .  $\triangleleft$

By the inductive assumption,  $G'$  admits a proper coloring  $f_2$  using colors  $[2(\omega - 1) - 1]$ . Finally, we define  $f: \mathcal{I} \rightarrow [2\omega - 1]$  as follows:

$$f(x) = \begin{cases} f_1(x) & \text{if } x \in R, \\ f_2(x) + 2 & \text{if } x \notin R. \end{cases}$$

We claim that  $f$  is a proper coloring of  $G$ . (For an example, see Figure 2.)

First, note that if  $x, y \in \mathcal{I}$  are distinct and  $x \cap y \neq \emptyset$ , then  $f(x) \neq f(y)$ . Indeed, if  $x, y \in R$ , then  $f(x) = f_1(x) \neq f_1(y) = f(y)$ . If  $x, y \notin R$ , then  $f(x) = f_2(x) + 2 \neq f_2(y) + 2 = f(y)$ . Finally, if, say,  $x \in R$  and  $y \notin R$ , then  $f(x) \in \{1, 2\}$  and  $f(y) \geq 3$ .

It remains to argue that the second condition in the definition of a proper coloring holds as well. For a contradiction, let  $x$  and  $y$  be distinct intervals and assume that  $x \subseteq y$  and  $f(y) > f(x)$ . Note that  $x \notin M(\mathcal{I})$  and thus  $x \notin R$ . This implies that  $f(x) \geq 3$ . Since we assumed that  $f(y) > f(x)$ , we have that  $f(y) > 3$ . Hence,  $y \notin R$ . However, by the inductive assumption, we have that  $f(x) = f_2(x) + 2 > f_2(y) + 2 = f(y)$ , which yields the desired contradiction. This completes the proof.  $\blacktriangleleft$

Observe that the proof of Theorem 2 can be easily transformed into an efficient algorithm, which yields the following corollary.

► **Corollary 5.** *There is a 2-approximation algorithm for coloring interval containment graphs properly. Given a set of  $n$  intervals, the algorithm runs in  $O(n \log n)$  time.*

**Proof.** For any graph  $G$ , we have  $\chi(G) \geq \omega(G)$ . Hence, the approximation factor follows directly from Theorem 2.

It remains to implement the constructive proof of Theorem 2 efficiently. Let  $\mathcal{I}$  be the given set of intervals. For each interval  $I$  in  $\mathcal{I}$ , let  $r_I$  be the right endpoint of  $I$ . We go through the intervals from left to right in several phases. In each phase, we use two colors, except possibly in the last phase where we may use only one color. For phase  $i$  with  $i \geq 1$ , we reserve the set  $colors(i) = \{2i - 1, 2i\}$ . We use an augmented balanced binary search tree  $\mathcal{T}$  to store the intervals in  $\mathcal{I}$ . We will query  $\mathcal{T}$  in two ways. A query of type Q1 in  $\mathcal{T}$  with a value  $x \in \mathbb{R} \cup \{-\infty\}$  will return, among all intervals whose left endpoint is at least  $x$ , one with leftmost left endpoint (and *nil* if such an interval does not exist). A query of type Q2 in  $\mathcal{T}$  with a value  $y \in \mathbb{R}$  will return, among all intervals whose left endpoint is at most  $y$ , one with rightmost right endpoint (and *nil* if such an interval does not exist). Note that the two queries are not symmetric.

Algorithm 1 describes our algorithm in pseudocode. Initially,  $\mathcal{T}$  stores all intervals in  $\mathcal{I}$ . The algorithm terminates once  $\mathcal{T}$  is empty and all intervals are colored.

■ **Algorithm 1** 2-APPROXIMATE-COLORING(set  $\mathcal{I}$  of  $n$  intervals).

---

```

 $\mathcal{T}$ .initialize( $\mathcal{I}$ )
 $i = 0$ 
while not  $\mathcal{T}$ .empty() do
     $i = i + 1$  // start new phase
     $I = \mathcal{T}$ .Q1( $-\infty$ )
     $c = I$ .color =  $2i - 1$  // color  $I$  and set current color
     $\mathcal{T}$ .remove( $I$ )
    while not  $\mathcal{T}$ .empty() do
         $I' = \mathcal{T}$ .Q2( $r_I$ )
        if  $I' == \text{nil}$  or  $r_{I'} \leq r_I$  then // no interval contains  $r_I$ 
             $I' = \mathcal{T}$ .Q1( $r_I$ )
            if  $I' == \text{nil}$  then break // finish current phase
             $I'$ .color =  $c$  // color  $I'$  and keep current color
        else //  $I$  and  $I'$  overlap
             $c = I'$ .color = colors( $i$ ) \ { $c$ } // color  $I'$  and swap current color
             $\mathcal{T}$ .remove( $I'$ )
             $I = I'$ 

```

---

We start each phase by Q1-querying  $\mathcal{T}$  with  $-\infty$ . This yields the leftmost interval  $I$  stored in  $\mathcal{T}$ . We color  $I$  with the smaller color  $2i - 1$  reserved for the current phase  $i$ . Let the *current color*  $c$  be this color. We remove  $I$  from  $\mathcal{T}$ . Then we Q2-query  $\mathcal{T}$  with the right endpoint  $r_I$  of  $I$  and consider the following two possibilities.

**Case I:** If the Q2-query returns *nil* or an interval that lies completely to the left of  $r_I$ , we Q1-query  $\mathcal{T}$  with  $r_I$  for an interval to the right of  $r_I$ . If such an interval  $I'$  exists, it must be disjoint from  $I$ , so we color  $I'$  with the current color  $c$ . Otherwise, we start a new phase.

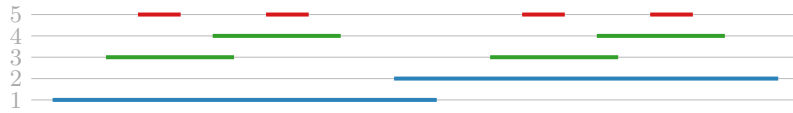
**Case II:** If the Q2-query returns an interval  $I'$  that overlaps with the previous interval  $I$ , we color  $I'$  with the other color  $c'$  that we reserved for the current phase, that is,  $\{c'\} = \text{colors}(i) \setminus \{c\}$ . Then we set the current color  $c$  to  $c'$ .

In either case, if we do not start a new phase, we remove  $I'$  from  $\mathcal{T}$  and proceed with the next Q2-query as above, with  $I'$  now playing the role of  $I$ .

It remains to implement the balanced binary search tree  $\mathcal{T}$ . The key of an interval is its left endpoint. For simplicity, we assume that the intervals are stored in the leaves of  $\mathcal{T}$  and that the key of each inner node is the maximum of the keys in its left subtree. This suffices to answer queries of type Q1. For queries of type Q2, we augment  $\mathcal{T}$  by storing, with each node  $\nu$ , a value  $\max(\nu)$  that we set to the maximum of the right endpoints among all intervals in the subtree rooted at  $\nu$ . (We also store a pointer  $\mu(\nu)$  to the interval that yields the maximum.) In a Q2-query with a value  $y$ , we search for the largest key  $k \leq y$ . Let  $\pi$  be the search path in  $\mathcal{T}$ , and initialize  $m$  with  $-\infty$ . When traversing  $\pi$ , we inspect each node  $\nu$  that hangs off  $\pi$  on the left side. If  $\max(\nu) > m$ , then we set  $m = \max(\nu)$  and  $\rho = \mu(\nu)$ . When we reach a leaf,  $\rho$  points to an interval whose right endpoint is maximum among all intervals whose left endpoint is at most  $y$ .

The runtime of  $O(n \log n)$  is obvious since we insert, query, and delete each interval in  $O(\log n)$  time exactly once. ◀





■ **Figure 3** Instance for the proof of Proposition 6 for  $n = 3$ :  $|\mathcal{I}_n| = 3 \cdot 2^{n-1} - 2 = 10$ ,  $\omega(\mathcal{I}_n) = n = 3$ , and  $\chi(\mathcal{I}_n) = 2n - 1 = 5$ .

► **Proposition 6.** *There is an infinite family  $(\mathcal{I}_n)_{n \geq 1}$  of sets of intervals with  $|\mathcal{I}_n| = 3 \cdot 2^{n-1} - 2$ ,  $\chi(\mathcal{C}[\mathcal{I}_n]) = 2n - 1$ , and  $\omega(\mathcal{C}[\mathcal{I}_n]) = n$ .*

**Proof.** The construction is iterative. The family  $\mathcal{I}_1$  consists of a single interval of unit length.

Now let  $n > 1$  and suppose that we have defined  $\mathcal{I}_{n-1}$  and want to define  $\mathcal{I}_n$ . We introduce two new intervals  $\ell_n$  and  $r_n$ , both of length  $3^{n-1}$ , that overlap slightly. Then we introduce two copies of  $\mathcal{I}_{n-1}$ . All intervals of one copy are contained in  $\ell_n \setminus r_n$ , and all intervals of the other copy are contained in  $r_n \setminus \ell_n$ .

The number of intervals in  $\mathcal{I}_n$  is given by the recursion:  $f(1) = 1$  and  $f(n) = 2f(n-1) + 2$ , which solves to  $f(n) = 3 \cdot 2^{n-1} - 2$ . Furthermore, it is straightforward to observe that with each step of the construction, the size of a largest clique increases by 1.

We claim that, for  $i \in [n]$ , in any proper coloring of  $\mathcal{C}[\mathcal{I}_i]$ , the difference between the largest and the smallest color used is at least  $2i - 2$ . Clearly, the claim holds for  $i = 1$ . Now assume that it holds for  $i = n - 1$ . Consider any proper coloring of  $\mathcal{C}[\mathcal{I}_n]$ , and let  $m$  be the minimum color used in this coloring. The colors of  $\ell_n$  and  $r_n$  must be different. Without loss of generality, suppose that the color of  $r_n$  is larger than the color of  $\ell_n$ . In particular, the color of  $r_n$  is at least  $m + 1$ . Now consider the copy of  $\mathcal{I}_{n-1}$  contained in  $r_n$ . The color of each interval in this copy must be larger than the color of  $r_n$ , so in particular the minimum color used for this copy of  $\mathcal{I}_{n-1}$  is at least  $m + 2$ . By the inductive assumption, some interval in this copy of  $\mathcal{I}_{n-1}$  receives a color that is at least  $m + 2 + 2(n - 1) - 2 = 2n - 2 + m$ . Summing up, the difference between the largest and the smallest color used for  $\mathcal{C}[\mathcal{I}_n]$  is at least  $2n - 2$ .

Given that the minimum color is 1, we conclude that  $\chi(\mathcal{C}[\mathcal{I}_n]) \geq 2n - 1$ .

For the upper bound, we color  $\mathcal{C}[\mathcal{I}_n]$  as follows. For  $n = 1$ , we color the only interval with color 1. For  $n > 1$ , we color  $\ell_n$  with color 1 and  $r_n$  with color 2. Next, for each of the two copies of  $\mathcal{I}_{n-1}$ , we use the proper coloring defined inductively with all colors increased by 2, see Figure 3. ◀

#### 4 Coloring Containment Interval Graphs Is NP-Hard

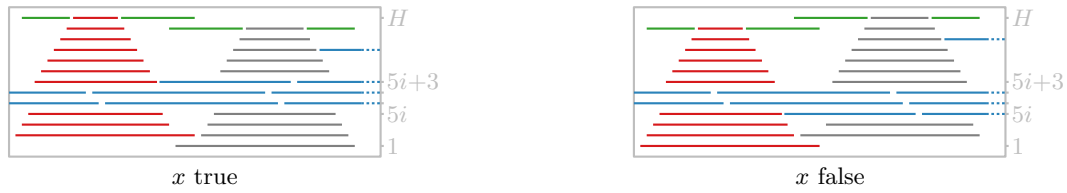
In this section we show that it is NP-hard to color a containment interval graph with a given number of colors.

► **Theorem 7.** *Given a set  $\mathcal{I}$  of intervals and a positive integer  $k$ , it is NP-hard to decide whether  $k$  colors suffice to color  $\mathcal{C}[\mathcal{I}]$ , that is, whether  $\chi(\mathcal{C}[\mathcal{I}]) \leq k$ .*

**Proof.** We describe a reduction from (exact) 3-SAT, i.e., the satisfiability problem where every clause contains exactly three literals. Let  $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$  be an instance of 3-SAT where, for each clause  $C_i$  ( $i \in [m]$ ), the literals are negated or unnegated variables from the set  $\{x_1, x_2, \dots, x_n\}$ , and let  $H = 5(m + 1)$  be a threshold.

Using  $\varphi$ , we construct in polynomial time a set of intervals (with pairwise distinct endpoints) such that the corresponding containment interval graph has a proper coloring with  $H$  colors if and only if  $\varphi$  is satisfiable. To this end, we introduce *variable gadgets*





**Figure 4** Variable gadget for the proof of Theorem 7 in its two states. The blue intervals with dots extend to the clause gadgets. The topmost blue interval (starting immediately to the right of a gray interval) indicates that  $x$  is part of a clause  $C_j$  with  $j > i$ ; the blue interval (with a small gap) that starts immediately to the right of a red interval indicates that  $\neg x$  is part of the clause  $C_i$ .

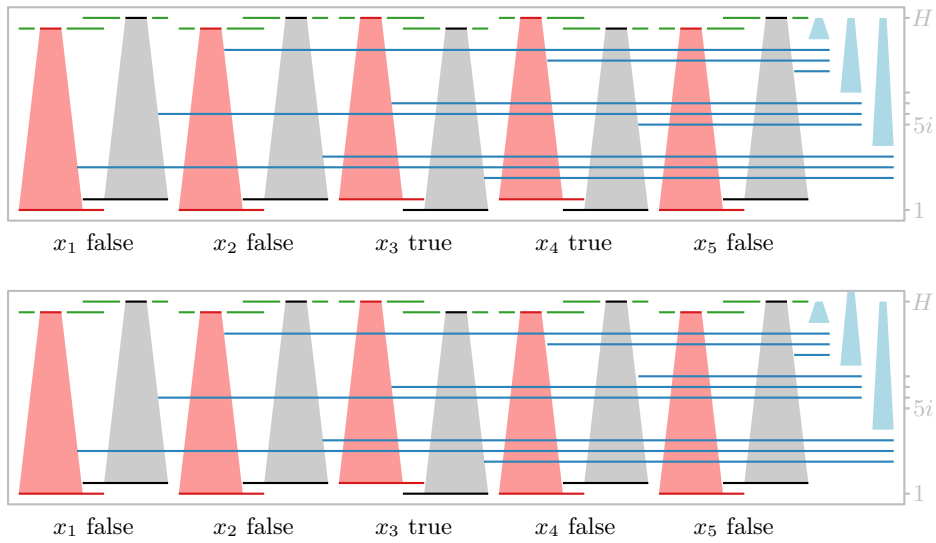
and *clause gadgets*, which are sets of intervals representing the variables and clauses of  $\varphi$ , respectively. Our main building structure used in these gadgets is a *Christmas tree*, that is, an ordered set of intervals where each interval contains its successor; see, for example, the set of red intervals in Figure 4. Clearly, the intervals of a Christmas tree form a totally ordered clique and any proper coloring needs to observe this order. In Figure 5, Christmas trees are represented by trapezoids. The *height* of a Christmas tree is the number of intervals it consists of.

Let  $j \in [n]$ . The variable gadget for  $x_j$  consists of two Christmas trees (formed by the red and gray intervals in Figure 4) whose longest intervals overlap and, for each tree, of two additional intervals (green in Figure 4). These green intervals lie immediately to the left and to the right of the shortest interval in their tree. The right green interval of the red tree overlaps the left green interval of the gray tree. Figure 4 depicts two representations of the same gadget for a variable  $x$ , each with its own coloring of the intervals (encoded by the height of the intervals; see the numbers at the right side of the gray box). The left representation with its coloring corresponds to assigning true to  $x$ , the right representation corresponds to assigning false. The height of the red tree is  $H - 1$  minus the number of occurrences of literals  $x_{j'}$  and  $\neg x_{j'}$  with  $j' < j$  in  $\varphi$ . The height of the gray tree is that of the red tree minus the number of occurrences of  $\neg x_j$  in  $\varphi$ . We say that  $x_j$  is set to true if the bottom interval of the gray tree has color 1; otherwise we say that  $x_j$  is set to false.

For  $i \in [m]$ , the gadget for clause  $C_i$  consists of a Christmas tree (light blue in Figure 5) of height  $H - (5i + 2) = 5(m - i) + 3$ . All clause gadgets are placed to the right of all variable gadgets, in the order  $C_1, \dots, C_m$  from left to right.

The key idea to transport a Boolean value from a variable gadget to a clause gadget is to add, for each occurrence of a literal  $\ell_j \in \{x_j, \neg x_j\}$  in a clause  $C_i$ , an “arm” (blue intervals in Figures 4 and 5) that ends to the right of the clause gadget (the light blue Christmas tree) corresponding to  $C_i$  and starts immediately to the right of the  $5i$ -th interval of the gray tree (if  $\ell_j = x_j$ ) or of the red tree (if  $\ell_j = \neg x_j$ ) corresponding to  $\ell_j$ . The arm is represented by a sequence of intervals that are separated by a small gap within each Christmas tree of each clause gadget passed by the arm (such that, for any two arms, their gaps are disjoint and the resulting intervals do not contain each other). Assuming that the total number of colors is  $H$ , two intervals of the same arm that are separated by a gap need to get the same color because, at the gap,  $H - 2$  colors are occupied by other intervals and the one remaining “wrong” color is blocked due to the green intervals of the variable gadgets; see Figures 4 and 5. The green intervals are contained by the blue intervals of the arms and need to get color  $H$  or  $H - 1$ .

If there is a satisfying truth assignment for  $\varphi$ , then there is a proper coloring that colors the variable gadgets such that they represent this truth assignment. As for every clause  $C_i$ , at least one of its literals in  $C_i$  is true, the corresponding arm can use color  $5i$ . Then, the



■ **Figure 5** Variable gadgets (red and gray) and clause gadgets (blue) for the 3-SAT instance  $(\neg x_2 \vee \neg x_4 \vee x_5) \wedge (x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee x_3)$  with a fulfilling truth assignment (above) and a non-fulfilling assignment (below). Note that the latter uses one color more (is one level higher).

arms of the other literals that occur in  $C_i$  can use colors  $5i + 1$  and  $5i + 2$ . This allows the light blue Christmas tree representing clause  $C_i$ , which has height  $H - 5i - 2$  and is contained in the rightmost interval of each of these arms, to use the colors  $\{5i + 3, 5i + 4, \dots, H\}$ .

Now suppose for a contradiction that there is no satisfying truth assignment for  $\varphi$ , but that there is a proper coloring with  $H$  colors. This coloring assigns a truth value to each variable gadget (depending on whether the bottommost interval of the red or the gray tree has color 1). Clearly, there is a clause  $C_i$  in  $\varphi$  that is not satisfied by this truth assignment. Hence, none of the arms connecting the clause gadget of  $C_i$  with its three corresponding variable gadgets can use color  $5i$ . Hence they must use colors  $5i + 1$ ,  $5i + 2$ , and  $5i + 3$  (or higher). This forces the (blue) Christmas tree representing clause  $C_i$  to use colors  $\{5i + 4, \dots, H, H + 1\}$ .

Thus, a proper coloring with  $H$  colors exists if and only if  $\varphi$  is satisfiable. ◀

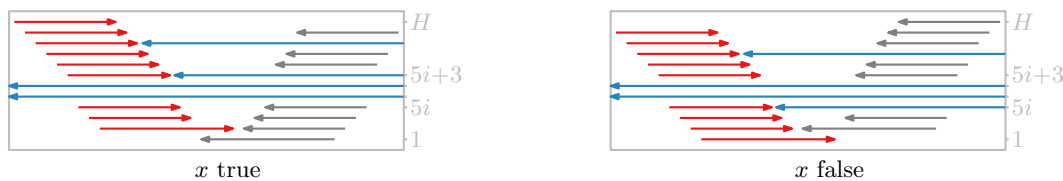
## 5 Coloring Bidirectional Interval Graphs Is NP-Hard

In this section we show that it is NP-hard to color a bidirectional interval graph with a given number of colors. For a set  $\mathcal{I}$  of intervals and a function  $o$  that maps every interval in  $\mathcal{I}$  to an orientation (left-going or right-going), let  $\mathcal{B}[\mathcal{I}, o]$  be the bidirectional interval graph induced by  $\mathcal{I}$  and  $o$ .

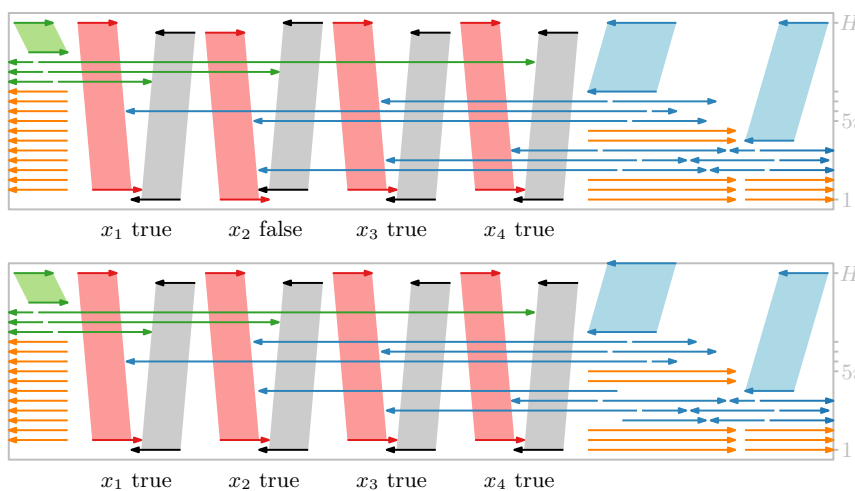
► **Theorem 8.** *Given a set  $\mathcal{I}$  of intervals with orientations  $o$  and a positive integer  $k$ , it is NP-hard to decide whether  $k$  colors suffice to color  $\mathcal{B}[\mathcal{I}, o]$ , that is, whether  $\chi(\mathcal{B}[\mathcal{I}, o]) \leq k$ .*

**Proof.** We use the same ideas as in the proof of Theorem 7, but now we reduce from MONOTONE 3-SAT, the version of 3-SAT where every clause contains only negated or only unnegated variables as literals. For an overview, see Figures 6 and 7. Let  $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$  be the given instance of MONOTONE 3-SAT with variables  $\{x_1, x_2, \dots, x_n\}$ . As before, let  $H = 5(m + 1)$  be the number of colors sufficient for coloring a yes-instance.

We now construct variable and clause gadgets by specifying a set  $\mathcal{I}$  of intervals with orientations  $o$ . Our intervals have pairwise distinct endpoints. Our main building structures are *left- and right-going staircases*. A left-going staircase (gray in Figures 6 and 7) is an



**Figure 6** Variable gadget for the proof of Theorem 8 in its two states. Intervals directions are indicated by arrow heads. The blue intervals extend (to the right) to the clause gadgets of only negated variables. The two blue arrow heads starting next to the red intervals indicate that the clause  $C_i$  and a clause  $C_j$  with  $j > i$  contain the literal  $\neg x$ .



**Figure 7** Variable gadgets (red and gray) and clause gadgets (light green and light blue) for the MONOTONE 3-SAT instance  $(x_1 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_4)$  with a fulfilling truth assignment (top) and a non-fulfilling assignment (bottom), which use  $H$  and  $H + 1$  colors, respectively. Interval directions are indicated by arrow heads.

ordered set of left-going intervals that share a common point and whose left and right endpoints are in the order of the set. A right-going staircase is symmetric (red in Figures 6 and 7). Observe that staircases in (bi)directional interval graphs behave like Christmas trees in containment graphs: they form totally ordered cliques. The *height* of a staircase is the number of its intervals. In Figure 7, we draw staircases as parallelograms and we indicate left- and right-going intervals by arrow heads.

Let  $i \in [m]$ . If the clause  $C_i$  has only negated literals, we again have three “arms” in  $C_i$  starting to the right of the  $5i$ -th interval of the red staircase of the three corresponding variables; see the blue intervals in Figures 6 and 7. The intervals of these arms are left-going and they are not split by a gap at the other variable gadgets this time because now we need to contain the left-going intervals of the staircases to have edges instead of arcs in  $\mathcal{B}[\mathcal{I}, o]$ . The arms end below a left-going light blue staircase (see Figure 7 on the upper right side) of height  $5(m - i) + 3$  whose maximum color is  $H$  if and only if none of the corresponding arms gets a color greater than  $5i + 2$ .

Note that we need to avoid arcs between the arms. Therefore, we let every arm have a gap below each blue staircase that it passes. These gaps do not overlap and their order is inverse to the order of the left endpoints of the involved intervals. We continue the arm with a right-going interval (to avoid an arc with the blue staircase and the other arms) ending to

the right of the blue staircase, where we continue again with a left-going interval. Consider such an arm with intervals  $I$  and  $I'$  (going in different directions) around a gap. At this gap, it is important that no color smaller than the color of  $I$  is available for  $I'$ , forcing  $I'$  to also get the color of  $I$ . Hence, we add long left-going intervals blocking every color not occupied by an arm or the blue staircase; see the orange intervals in Figure 7.

For every clause with only unnegated literals, we use the same construction but mirrored (connecting to the gray staircases); see the green intervals and light green staircases depicted in the top left corner of Figure 7.

We now have the same conditions as in the proof of Theorem 7: If there is a satisfying truth assignment for  $\varphi$ , there is a proper coloring, which colors the variable gadgets such that they represent this truth assignment. Again, for every  $i \in [m]$ , clause  $C_i$  contains at least one literal set to true. Hence, the corresponding arm can get color  $5i$ , and the other two arms and the (light blue or light green) staircase of  $C_i$  get colors  $\{5i + 1, 5i + 2, \dots, H\}$ .

If there is no satisfying truth assignment for  $\varphi$ , then there is a clause  $C_i$  none of whose arms (as a whole) can get color  $5i$ . If each arm occupies only one layer, then an interval of the clause gadget of  $C_i$  requires color  $H + 1$ . If there is an arm occupying more than one layer, then below a clause gadget of some clause  $C_{i'}$ , there are two colors blocked by this arm (at one of its gaps). Then, however, an interval of the clique of size  $H - 1$  at this gap belonging to the (light blue or light green) staircase of  $C_{i'}$ , to the other arms, or to the orange “blocker” intervals requires color  $H + 1$ ; see Figure 7 for such an example. ◀

## 6 Coloring General Mixed Interval Graphs

In this section we consider a further generalization of mixed interval graphs. We are dealing with an interval graph  $G$  whose edges can be arbitrarily oriented (or stay undirected). In other words, the edge directions are not related to the geometry of the intervals.

Observe that a proper coloring of  $G$  exists if and only if  $G$  does not contain a directed cycle. Let  $\chi(G)$  denote the minimum number of colors in a proper coloring of  $G$ , if it exists, or  $\infty$  otherwise. We point out that the existence of a directed cycle can be determined in polynomial time (using, for example, depth-first search).

Note that clearly we have  $\omega(G) \leq \chi(G)$ . However, there is another parameter that enforces a large chromatic number even in sparse graphs. A *directed path* (of length  $t$ ) in  $G$  is a sequence of vertices  $\langle v_1, v_2, \dots, v_{t+1} \rangle$ , such that, for each  $i \in [t]$ , the arc  $(v_i, v_{i+1})$  exists. Let  $\lambda(G)$  denote the length of a longest directed path in  $G$ .

Note that the vertices in a directed path receive pairwise distinct colors in any proper coloring. Thus we have  $\chi(G) \geq \lambda(G) + 1$ , and consequently  $\chi(G) \geq \max\{\omega(G), \lambda(G) + 1\}$ .

► **Theorem 9.** *Let  $G$  be a mixed interval graph without directed cycles. Then  $\chi(G) \leq (\lambda(G) + 1) \cdot \omega(G)$ .*

**Proof.** Let  $V$  denote the vertex set of  $G$ . Let  $G^\rightarrow$  be the graph obtained from  $G$  by removing all edges. Clearly,  $G^\rightarrow$  is a DAG. We partition  $V$  into *layers*  $L_0, L_1, \dots$  as follows. The set  $L_0$  consists of the vertices that are sources in  $G^\rightarrow$ , i.e., they do not have incoming arcs. Then, for  $i = 1, 2, \dots$ , we iteratively define  $L_i$  to be the set of sources in  $G^\rightarrow \setminus \bigcup_{j=0}^{i-1} L_j$ . Note that  $\lambda(G) = \max\{i : L_i \neq \emptyset\}$ . For  $x \in V$ , let  $\ell(x) \in \{0, \dots, \lambda(G)\}$  denote the unique  $i$  such that  $x \in L_i$ .

Recall that the underlying undirected graph of  $G$ ,  $U(G)$ , is an (undirected) interval graph, and thus  $\chi(U(G)) = \omega(U(G)) = \omega(G)$ . Let  $c : V \rightarrow [\omega(U(G))]$  be an optimal proper coloring of  $U(G)$ .



■ **Figure 8** For any  $k \geq 1$ , the set  $\mathcal{I}_k \cup \mathcal{I}'_k$  of intervals gives rise to a mixed interval graph  $G_k$  with  $2k^2$  vertices,  $\lambda(G_k) = k - 1$ ,  $\omega(G_k) = 2k$ , and  $\chi(G_k) = (k + 1) \cdot k = (\lambda(G_k) + 2) \cdot \omega(G_k)/2$ .

Now we define a coloring  $f$  of  $G$ : for  $x \in V$ , let  $f(x) = \ell(x) \cdot \omega(G) + c(x)$ . Note that  $1 \leq f(x) \leq (\lambda(G) + 1) \cdot \omega(G)$ . We claim that  $f$  is a proper coloring.

Consider an edge  $\{x, y\}$ . As this is also an edge in  $U(G)$ , we obtain that  $c(x) \neq c(y)$ , and so  $f(x) \neq f(y)$ . Now consider an arc  $(x, y)$ . Its existence implies that  $\ell(x) < \ell(y)$ , and thus  $f(x) < f(y)$ . ◀

For some instances, the above bound is asymptotically tight.

► **Proposition 10.** *There is an infinite family  $(G_k)_{k \geq 1}$  of mixed interval graphs with  $|V(G_k)| = 2k^2$ ,  $\lambda(G_k) = k - 1$ ,  $\omega(G_k) = 2k$ , and  $\chi(G_k) = (k + 1) \cdot k = (\lambda(G_k) + 2) \cdot \omega(G_k)/2$ .*

**Proof.** Let  $\mathcal{I}_k = \mathcal{I}_{k,1} \cup \mathcal{I}_{k,2} \cup \dots \cup \mathcal{I}_{k,k}$  be a set of  $k^2$  intervals defined as follows; see Figure 8 for  $\mathcal{I}_4$ . For  $i \in [k]$ , let  $\mathcal{I}_{k,i}$  be a multiset that contains  $k$  copies of the interval  $[6i, 6i + 8]$ . Similarly, let  $\mathcal{I}'_k = \mathcal{I}'_{k,1} \cup \mathcal{I}'_{k,2} \cup \dots \cup \mathcal{I}'_{k,k}$  be a set of  $k^2$  intervals such that  $\mathcal{I}'_k$  is the image of mirroring  $\mathcal{I}_k$  at the point  $x = 6k + 7$ . Note that, for  $i \in [k - 1]$ , every interval in  $\mathcal{I}_{k,i}$  intersects every interval in  $\mathcal{I}_{k,i+1}$  and every interval in  $\mathcal{I}'_{k,i}$  intersects every interval in  $\mathcal{I}'_{k,i+1}$ . Additionally, every interval in  $\mathcal{I}_{k,k}$  intersects every interval in  $\mathcal{I}'_{k,k}$ .

Let  $G_k$  be a mixed interval graph for the set  $\mathcal{I}_k \cup \mathcal{I}'_k$ . We direct the edges of  $G_k$  as follows. Let  $\{I, I'\}$  be a pair of intervals in  $\mathcal{I}_k \cup \mathcal{I}'_k$  that intersect each other. If  $I$  and  $I'$  are copies of the same interval, then  $\{I, I'\}$  is an edge of  $G_k$ . Otherwise,  $(I, I')$  is an arc of  $G_k$  if  $\{I, I'\} \subseteq \mathcal{I}_k$  and  $I$  lies further to the left than  $I'$ , if  $\{I, I'\} \subseteq \mathcal{I}'_k$  and  $I$  lies further to the right than  $I'$ , or if  $(I, I') \in \mathcal{I}_{k,k} \times \mathcal{I}'_{k,k}$ . It is easy to see that  $G_k$  has the desired properties. ◀

Note that the mixed interval graphs that we constructed in the proof above are even *directional* interval graphs.

## 7 Open Problems

The obvious open problems are improvements to the results in Table 1, in particular: Is there a constant-factor approximation algorithm for coloring general mixed interval graphs? For applications in graph drawing, a better-than-2 approximation for coloring bidirectional interval graphs is of particular interest.

Is there a linear-time recognition algorithm for directional or containment interval graphs? Is there a polynomial-time recognition algorithm for bidirectional interval graphs?

Using a reduction from MAX-3-SAT instead of 3-SAT, it may be possible to adjust our NP-hardness proofs in order to show APX-hardness. To this end, the difference in the number of colors needed to color a yes-instance and the number of colors needed to color a no-instance would have to be proportional to the number of clauses that cannot be satisfied. We were not able to force such a large difference, hence we leave the APX-hardness of (or the existence of a PTAS for) coloring containment and birectional interval graphs open.

## References

- 1 Matthias Beck, Daniel Blado, Joseph Crawford, Taïna Jean-Louis, and Michael Young. Mixed graph colorings. In *Proc. Sci. Nat. Conf. of the Society for Advancement of Hispanics/Chicanos and Native Americans*, 2012.
- 2 Matthias Beck, Daniel Blado, Joseph Crawford, Taïna Jean-Louis, and Michael Young. On weak chromatic polynomials of mixed graphs. *Graphs Combin.*, 31:91–98, 2015. doi:10.1007/s00373-013-1381-1.
- 3 Peter Brucker. *Scheduling Algorithms*. Springer, 5 edition, 1995. doi:10.1007/978-3-540-69516-5.
- 4 Dominique de Werra. Restricted coloring models for timetabling. *Discrete Math.*, 165–166:161–170, 1997. doi:10.1016/S0012-365X(96)00208-7.
- 5 Hanna Furmańczyk, Adrian Kosowski, and Paweł Żyliński. A note on mixed tree coloring. *Inf. Process. Lett.*, 106(4):133–135, 2008. doi:10.1016/j.ipl.2007.11.003.
- 6 Hanna Furmańczyk, Adrian Kosowski, and Paweł Żyliński. Scheduling with precedence constraints: Mixed graph coloring in series-parallel graphs. In *Proc. PPAM'07*, pages 1001–1008, 2008. doi:10.1007/978-3-540-68111-3\_106.
- 7 Grzegorz Gutowski, Konstanty Junosza-Szaniawski, Felix Klesen, Paweł Rzażewski, Alexander Wolff, and Johannes Zink. Coloring and recognizing directed interval graphs. ArXiv report, 2023. doi:10.48550/arXiv.2303.07960.
- 8 Grzegorz Gutowski, Florian Mittelstädt, Ignaz Rutter, Joachim Spoerhase, Alexander Wolff, and Johannes Zink. Coloring mixed and directional interval graphs. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Proc. 30th Int. Symp. Graph Drawing & Network Vis. (GD'22)*, volume 13764 of *LNCS*, pages 418–431. Springer, 2023. doi:10.1007/978-3-031-22203-0\_30.
- 9 Pierre Hansen, Julio Kuplinsky, and Dominique de Werra. Mixed graph colorings. *Math. Methods Oper. Res.*, 45:145–160, 1997. doi:10.1007/BF01194253.
- 10 George S. Lueker and Kellogg S. Booth. A linear time algorithm for deciding interval graph isomorphism. *J. ACM*, 26(2):183–195, 1979. doi:10.1145/322123.322125.
- 11 Ross M. McConnell and Jeremy P. Spinrad. Modular decomposition and transitive orientation. *Discrete Math.*, 201(1):189–241, 1999. doi:10.1016/S0012-365X(98)00319-7.
- 12 Bernard Ries and Dominique de Werra. On two coloring problems in mixed graphs. *Eur. J. Comb.*, 29(3):712–725, 2008. doi:10.1016/j.ejc.2007.03.006.
- 13 Paolo Serafini and Walter Ukovich. A mathematical model for the fixed-time traffic control problem. *Europ. J. Oper. Res.*, 42(2):152–165, 1989. doi:10.1016/0377-2217(89)90318-4.
- 14 Yuri N. Sotskov. Mixed graph colorings: A historical review. *Mathematics*, 8(3):385:1–24, 2020. doi:10.3390/math8030385.
- 15 Yuri N. Sotskov, Vjacheslav S. Tanaev, and Frank Werner. Scheduling problems and mixed graph colorings. *Optimization*, 51(3):597–624, 2002. doi:10.1080/0233193021000004994.
- 16 Yuri N. Sotskov and Vyacheslav S. Tanaev. Chromatic polynomial of a mixed graph. *Vestsi Akademii Navuk BSSR. Seryya Fizika-Matematychnykh Navuk*, 6:20–23, 1976.
- 17 Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.*, 11(2):109–125, 1981. doi:10.1109/TSMC.1981.4308636.
- 18 Vjacheslav S. Tanaev, Yuri N. Sotskov, and V.A. Strusevich. *Scheduling Theory: Multi-Stage Systems*. Kluwer Academic Publishers, 1994.
- 19 Johannes Zink, Julian Walter, Joachim Baumeister, and Alexander Wolff. Layered drawing of undirected graphs with generalized port constraints. *Comput. Geom.*, 105–106(101886):1–29, 2022. doi:10.1016/j.comgeo.2022.101886.