The Recognition of Simple-Triangle Graphs and of Linear-Interval Orders Is Polynomial*

George B. Mertzios

School of Engineering and Computing Sciences, Durham University, UK
george.mertzios@durham.ac.uk

Abstract. Intersection graphs of geometric objects have been extensively studied, both due to their interesting structure and their numerous applications; prominent examples include interval graphs and permutation graphs. In this paper we study a natural graph class that generalizes both interval and permutation graphs, namely *simple-triangle* graphs. Simple-triangle graphs – also known as *PI* graphs (for Point-Interval) – are the intersection graphs of triangles that are defined by a point on a line L_1 and an interval on a parallel line L_2. They lie naturally between permutation and trapezoid graphs, which are the intersection graphs of line segments between L_1 and L_2 and of trapezoids between L_1 and L_2, respectively. Although various efficient recognition algorithms for permutation and trapezoid graphs are well known to exist, the recognition of simple-triangle graphs has remained an open problem since their introduction by Corneil and Kamula three decades ago. In this paper we resolve this problem by proving that simple-triangle graphs can be recognized in polynomial time. As a consequence, our algorithm also solves a longstanding open problem in the area of partial orders, namely the recognition of *linear-interval orders*, i.e. of partial orders $P = P_1 \cap P_2$, where P_1 is a linear order and P_2 is an interval order. This is one of the first results on recognizing partial orders P that are the intersection of orders from two different classes $\mathcal{P}_1$ and $\mathcal{P}_2$. In contrast, partial orders P which are the intersection of orders from the same class $\mathcal{P}$ have been extensively investigated, and in most cases the complexity status of these recognition problems has been established.

Keywords: Intersection graphs, PI graphs, recognition problem, partial orders, polynomial algorithm.

1 Introduction

A graph G is the *intersection graph* of a family $\mathcal{F}$ of sets if we can bijectively assign sets of $\mathcal{F}$ to vertices of G such that two vertices of G are adjacent if and only if the corresponding sets have a non-empty intersection. It turns out that many graph classes with important applications can be described as intersection

* This work was partially supported by (i) the EPSRC Grant EP/K022660/1 and (ii) the EPSRC Grant EP/G043434/1.

H.L. Bodlaender and G.F. Italiano (Eds.): ESA 2013, LNCS 8125, pp. 719–730, 2013.

graphs of set families that are derived from some kind of geometric configuration. One of the most prominent examples is that of *interval* graphs, i.e. the intersection graphs of intervals on the real line, which have natural applications in several fields, including bioinformatics and involving the physical mapping of DNA and the genome reconstruction[1] [3, 6, 7].

Generalizing the intersections on the real line, consider two parallel horizontal lines on the plane, L_1 (the upper line) and L_2 (the lower line). A graph G is a *simple-triangle* graph if it is the intersection graph of triangles that have one endpoint on L_1 and the other two on L_2. Furthermore, G is a *triangle* graph if it is the intersection graph of triangles with endpoints on L_1 and L_2, but now there is no restriction on which line contains one endpoint of every triangle and which contains the other two. Simple-triangle and triangle graphs are also known as PI and PI^* graphs, respectively [2, 4, 14], where PI stands for "Point-Interval". Such representations of simple-triangle and of triangle graphs are called *simple-triangle* (or PI) and *triangle* (or PI^*) *representations*, respectively. Simple-triangle and triangle graphs lie naturally between *permutation* graphs (i.e. the intersection graphs of line segments with one endpoint on L_1 and one on L_2) and *trapezoid* graphs (i.e. the intersection graphs of trapezoids with one interval on L_1 and the opposite interval on L_2) [2, 14]. Note that, using the notation PI for simple-triangle graphs, permutation graphs are PP (for "Point-Point") graphs, while trapezoid graphs are II (for "Interval-Interval") graphs [4].

A *partial order* is a pair $P = (U, R)$, where U is a finite set and R is an irreflexive transitive binary relation on U. Whenever $(x, y) \in R$ for two elements $x, y \in U$, we write $x <_P y$. If $x <_P y$ or $y <_P x$, then x and y are *comparable*, otherwise they are *incomparable*. P is a *linear order* if every pair of elements in U are comparable. Furthermore, P is an *interval order* if each element $x \in U$ is assigned to an interval I_x on the real line such that $x <_P y$ if and only if I_x lies completely to the left of I_y. One of the most fundamental notions on partial orders is *dimension*. For any partial order P and any class $\mathcal{P}$ of partial orders (e.g. linear order, interval order, semiorder, etc.), the $\mathcal{P}$*-dimension* of P is the smallest k such that P is the intersection of k orders from $\mathcal{P}$. In particular, when $\mathcal{P}$ is the class of linear orders, the $\mathcal{P}$-dimension of P is known as the *dimension* of P. Although in most cases we can efficiently recognize whether a partial order belongs to a class $\mathcal{P}$, this is not the case for higher dimensions. Due to a classical result of Yannakakis [15], it is NP-complete to decide whether the dimension, or the interval dimension, of a partial order is at most k, where $k \geq 3$.

There is a natural correspondence between graphs and partial orders. For a partial order $P = (U, R)$, the *comparability* (resp. *incomparability*) *graph* $G(P)$ of P has elements of U as vertices and an edge between every pair of comparable (resp. incomparable) elements. A graph G is a *(co)comparability graph* if G is the (in)comparability graph of a partial order P. There has been a long

[1] Benzer [1] earned the prestigious Lasker Award (1971) and Crafoord Prize (1993) partly for showing that the set of intersections of a large number of fragments of genetic material in a virus form an interval graph.

line of research in order to establish the complexity of recognizing partial orders of $\mathcal{P}$-dimension at most 2 (e.g. where $\mathcal{P}$ is linear orders [14] or interval orders [9]). In particular, since permutation (resp. trapezoid) graphs are the incomparability graphs of partial orders with dimension (resp. interval dimension) at most 2 [5,14], permutation and trapezoid graphs can be recognized efficiently by the corresponding partial order algorithms [9, 14].

In contrast, not much is known so far for the recognition of partial orders P that are the intersection of orders from different classes $\mathcal{P}_1$ and $\mathcal{P}_2$. One of the longstanding open problems in this area is the recognition of *linear-interval orders* P, i.e. of partial orders $P = P_1 \cap P_2$, where P_1 is a linear order and P_2 is an interval order. In terms of graphs, this problem is equivalent to the recognition of simple-triangle (i.e. PI) graphs, since PI graphs are the incomparability graphs of linear-interval orders; this problem is well known and remains open since the introduction of PI graphs in 1987 [4] (cf. for instance the books [2, 14]).

Our Contribution. In this article we establish the complexity of recognizing simple-triangle (PI) graphs, and therefore also the complexity of recognizing linear-interval orders. Given a graph G with n vertices, such that its complement $\overline{G}$ has m edges, we provide an algorithm with running time $O(n^2m)$ that either computes a PI representation of G, or it announces that G is not a PI graph. Equivalently, given a partial order $P = (U, R)$ with $|U| = n$ and $|R| = m$, our algorithm either computes in $O(n^2m)$ time a linear order P_1 and an interval order P_2 such that $P = P_1 \cap P_2$, or it announces that such orders P_1, P_2 do not exist. Surprisingly, it turns out that the seemingly small difference in the definition of simple-triangle (PI) graphs and triangle (PI*) graphs results in a very different behavior of their recognition problems; only recently it has been proved that the recognition of triangle graphs is NP-complete [11]. In addition, our polynomial time algorithm is in contrast to the recognition problems for the related classes of *bounded tolerance* (i.e. *parallelogram*) graphs [12] and of *max-tolerance* graphs [8], which have already been proved to be NP-complete.

As the main tool for our algorithm we introduce the notion of a *linear-interval cover* of bipartite graphs. As a second tool we identify a new tractable subclass of 3SAT, called *gradually mixed* formulas, for which we provide a linear time algorithm. The class of gradually mixed formulas is *hybrid*, i.e. it is characterized by both *relational* and *structural* restrictions on the clauses. Then, using the notion of a linear-interval cover, we are able to reduce our problem to the satisfiability problem of gradually mixed formulas.

Our algorithm proceeds as follows. First, it computes from the given graph G a bipartite graph $\widetilde{G}$, such that G is a PI graph if and only if $\widetilde{G}$ has a linear-interval cover. Second, it computes a gradually mixed Boolean formula ϕ such that ϕ is satisfiable if and only if $\widetilde{G}$ has a linear-interval cover. This formula ϕ can be written as $\phi = \phi_1 \wedge \phi_2$, where every clause of ϕ_1 has 3 literals and every clause of ϕ_2 has 2 literals. The construction of ϕ_1 and ϕ_2 is based on the fact that a necessary condition for $\widetilde{G}$ to admit a linear-interval cover is that its edges can be colored with two different colors (according to some restrictions). Then the edges of $\widetilde{G}$ correspond to literals of ϕ, while the two edge colors encode the truth

value of the corresponding variables. Furthermore every clause of ϕ_1 corresponds to the edges of an *alternating cycle* in $\widetilde{G}$ (i.e. a closed walk that alternately visits edges and non-edges) of length 6, while the clauses of ϕ_2 correspond to specific pairs of edges of $\widetilde{G}$ that are not allowed to receive the same color. Finally, the equivalence between the existence of a linear-interval cover of $\widetilde{G}$ and a satisfying truth assignment for ϕ allows us to use our linear algorithm to solve satisfiability on gradually mixed formulas in order to complete our recognition algorithm.

Organization of the Paper. We present in Section 2 the class of gradually mixed formulas and a linear time algorithm to solve satisfiability on this class. In Section 3 we provide the necessary notation and preliminaries on alternating cycles. Then in Section 4 we introduce the notion of a linear-interval cover of bipartite graphs to characterize PI graphs, and in Section 5 we translate the linear-interval cover problem to the satisfiability problem on a gradually mixed formula. Finally, in Section 6 we present our PI graph recognition algorithm.

2 A Tractable Subclass of 3SAT

In this section we introduce the class of *gradually mixed* formulas and we provide a linear time algorithm for solving satisfiability on this class. Any gradually mixed formula ϕ is a mix of binary and ternary clauses. That is, there exist a 3-CNF formula ϕ_1 (i.e. a formula in conjunctive normal form with at most 3 literals per clause) and a 2-CNF formula ϕ_2 (i.e. with at most 2 literals per clause) such that $\phi = \phi_1 \wedge \phi_2$, while ϕ satisfies some constraints among its clauses. Before we define gradually mixed formulas (cf. Definition 2), we first define *dual* clauses.

Definition 1. *Let ϕ_1 be a 3-CNF formula. If $\alpha = (\ell_1 \vee \ell_2 \vee \ell_3)$ is a clause of ϕ_1, then the $\overline{\alpha} = (\overline{\ell_1} \vee \overline{\ell_2} \vee \overline{\ell_3})$ is the* dual *clause of α.*

Note by Definition 1 that, whenever α is a clause of a formula ϕ_1, the dual clause $\overline{\alpha}$ of α may belong, or may not belong, to ϕ_1.

Definition 2. *Let ϕ_1 and ϕ_2 be CNF formulas with 3 literals and 2 literals in each clause, respectively. The mixed formula $\phi = \phi_1 \wedge \phi_2$ is* gradually mixed *if the next two conditions are satisfied:*

1. *Let α and β be two clauses of ϕ_1. Then α does not share exactly one literal with either the clause β or the clause $\overline{\beta}$.*
2. *Let $\alpha = (\ell_1 \vee \ell_2 \vee \ell_3)$ be a clause of ϕ_1. Then:*
   - *if $(\ell_0 \vee \ell_1)$ is a clause of ϕ_2, then ϕ_2 contains also (at least) one of the clauses $\{(\ell_0 \vee \overline{\ell_2}), (\ell_0 \vee \overline{\ell_3})\}$,*
   - *if $(\ell_0 \vee \overline{\ell_1})$ is a clause of ϕ_2, then ϕ_2 contains also (at least) one of the clauses $\{(\ell_0 \vee \ell_2), (\ell_0 \vee \ell_3)\}$.*

As an example of a gradually mixed formula, consider the formula $\phi = \phi_1 \wedge \phi_2$, where $\phi_1 = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (x_5 \vee x_6 \vee \overline{x_7})$ and $\phi_2 = (x_8 \vee \overline{x_3}) \wedge (x_8 \vee x_1) \wedge (x_8 \vee x_4) \wedge (\overline{x_8} \vee x_9) \wedge (x_5 \vee x_{10}) \wedge (\overline{x_6} \vee x_{10})$.

Note by Definition 2 that the class of gradually mixed formulas contains 2SAT as a proper subclass, since every 2-CNF formula ϕ_2 can be written as a gradually mixed formula $\phi = \phi_1 \wedge \phi_2$ where $\phi_1 = \emptyset$. Furthermore the class of gradually mixed formulas ϕ is a *hybrid* class, since the conditions of Definition 2 concern simultaneously *relational* restrictions (i.e. where the clauses are restricted to be of certain types) and *structural* restrictions (i.e. where there are restrictions on how different clauses interact with each other). The intuition for the term *gradually mixed* in Definition 2 is that, whenever the sub-formulas ϕ_1 and ϕ_2 share more variables, the number of clauses of ϕ_2 that are imposed by condition 2 of Definition 2 increases. In the next theorem we use resolution to prove that satisfiability can be solved in linear time on gradually mixed formulas.

Theorem 1. *There exists a linear time algorithm which decides whether a given gradually mixed formula ϕ is satisfiable and computes a satisfying truth assignment of ϕ, if one exists.*

The conditions of Definition 2 which guarantee the tractability of gradually mixed formulas are *minimal*, in the sense that, if we remove any of these two conditions, the resulting subclass of 3SAT is NP-complete.

3 Preliminaries

Notation. In this article we consider finite, simple, and undirected graphs. An edge between two vertices u and v of a graph $G = (V, E)$ is denoted by uv, and in this case u and v are said to be *adjacent*. The *neighborhood* of a vertex $u \in V$ is the set $N(u) = \{v \in V \mid uv \in E\}$ of its adjacent vertices. The complement of G is denoted by $\overline{G}$, i.e. $\overline{G} = (V, \overline{E})$, where $uv \in \overline{E}$ if and only if $uv \notin E$. For any subset $E_0 \subseteq E$ of the edges of G, we denote for simplicity $G - E_0 = (V, E \setminus E_0)$. A subset $S \subseteq V$ of its vertices induces an *independent set* in G if $uv \notin E$ for every pair of vertices $u, v \in S$. Furthermore, S induces a *clique* in G if $uv \in E$ for every pair $u, v \in S$. A graph G is a *split graph* if its vertices can be partitioned into a clique K and an independent set I.

The smallest k for which there exists a proper k-coloring of G is the *chromatic number* of G, denoted by $\chi(G)$. If $\chi(G) = 2$ then G is a *bipartite* graph, i.e. its vertices are partitioned into two independent sets, the *color classes*. A bipartite graph G is denoted by $G = (U, V, E)$, where U and V are its color classes and E is the set of edges between them. For a bipartite graph $G = (U, V, E)$, its *bipartite complement* is the graph $\widehat{G} = (U, V, \widehat{E})$, where for two vertices $u \in U$ and $v \in V$, $uv \in \widehat{E}$ if and only if $uv \notin E$. A bipartite graph $G = (U, V, E)$ is a *chain graph* if the vertices of each color class can be ordered by inclusion of their neighborhoods, i.e. $N(u) \subseteq N(v)$ or $N(v) \subseteq N(u)$ for any two vertices u, v in the same color class. Note that chain graphs are closed under bipartite complementation, i.e. G is a chain graph if and only if $\widehat{G}$ is a chain graph.

For two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, we denote $G_1 \subseteq G_2$ whenever $E_1 \subseteq E_2$. Moreover, we denote for simplicity by $G_1 \cup G_2$ and $G_1 \cap G_2$ the graphs $(V, E_1 \cup E_2)$ and $(V, E_1 \cap E_2)$, respectively. Similarly, for any two partial

orders $P_1 = (U, R_1)$ and $P_2 = (U, R_2)$, we denote $P_1 \subseteq P_2$ whenever $R_1 \subseteq R_2$. Moreover, we denote for simplicity $P_1 \cup P_2$ and $P_1 \cap P_2$ for the partial orders $(U, R_1 \cup R_2)$ and $(U, R_1 \cap R_2)$, respectively.

Alternating Cycles in a Graph. The next definition of an *alternating cycle* is crucial for our recognition algorithm for PI graphs.

Definition 3. *Let $G = (V, E)$ be a graph, $\widetilde{E} \subseteq E$ be an edge subset, and $k \geq 2$. A set of $2k$ (not necessarily distinct) vertices $v_1, v_2, \ldots, v_{2k} \in V$ builds an* alternating cycle AC_{2k} in $\widetilde{E}$*, if $v_i v_{i+1} \in \widetilde{E}$ whenever i is even and $v_i v_{i+1} \notin E$ whenever i is odd (where indices are* mod $2k$*). Furthermore, we say that G has an alternating cycle AC_{2k}, whenever G has an AC_{2k} in the edge set $\widetilde{E} = E$.*

For instance, for $k = 3$, there exist two different possibilities for an AC_6, which are illustrated in Figures 1(a) and 1(b). These two types of an AC_6 are called an *alternating path of length* 5 or *of length* 6, respectively (AP_5 and AP_6 for short, respectively). Furthermore, note that for $k = 2$, a set of four vertices $v_1, v_2, v_3, v_4 \in V$ builds an alternating cycle AC_4 if $v_1v_2, v_3v_4 \notin E$ and $v_2v_3, v_1v_4 \in E$. There are three possible graphs on four vertices that build an alternating cycle, AC_4 which are illustrated in Figures 1(c)-1(e).

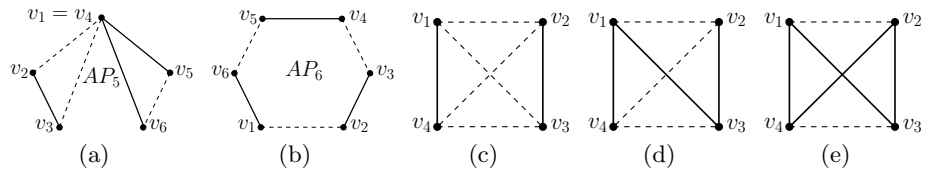

Fig. 1. The two possibilities for an AC_6: (a) an alternating path AP_5 of length 5 and (b) an alternating path AP_6 of length 6. Furthermore, the three possibilities for an AC_4: (c) a $2K_2$, (d) a P_4, and (e) a C_4. The solid lines denote edges of the graph and the dashed lines denote non-edges of the graph.

Alternating cycles can be used to characterize chain graphs as the bipartite graphs with no induced $2K_2$ [10]. We define now for any bipartite graph G the associated split graph of G, which we use extensively throughout of the paper.

Definition 4. *Let $G = (U, V, E)$ be a bipartite graph. The* associated split graph *of G is the split graph $H_G = (U \cup V, E')$, where $E' = E \cup (V \times V)$, i.e. H_G is the split graph made by G by replacing the independent set V of G by a clique.*

The next two definitions of a *conflict* between two edges and the *conflict graph* are essential for our results.

Definition 5. *Let $G = (V, E)$ be a graph and $e_1, e_2 \in E$. If the vertices of e_1 and e_2 build an AC_4 in G, then e_1 and e_2 are in* conflict*, and in this case we denote $e_1||e_2$ in G. Furthermore, an edge $e \in E$ is* committed *if there exists an edge $e' \in E$ such that $e||e'$; otherwise e is* uncommitted*.*

Definition 6 ([13]). *Let $G = (V, E)$ be a graph. The* conflict graph $G^* = (V^*, E^*)$ of G *is defined by*

- $V^* = E$ *and*
- *for every $e_1, e_2 \in E$, $e_1e_2 \in E^*$ if and only if $e_1||e_2$ in G.*

4 Linear-Interval Covers of Bipartite Graphs

In this section we introduce the crucial notion of a *linear-interval cover* of bipartite graphs (cf. Definition 9). Then we use linear-interval covers to provide a new characterization of PI graphs (cf. Theorem 3), which is one of the main tools for our PI graph recognition algorithm. First we provide in the next theorem the characterization of PI graphs using linear orders and interval orders.

Theorem 2. *Let $G = (V, E)$ be a cocomparability graph and P be a partial order of $\overline{G}$. Then G is a PI graph if and only if $P = P_1 \cap P_2$, where P_1 is a linear order and P_2 is an interval order.*

For every partial order P we define now the domination bipartite graph $C(P)$, which has been used to characterize interval orders [9]. Here "C" stands for "Comparable", since the definition of $C(P)$ uses the comparable elements of P.

Definition 7 ([9]). *Let $P = (U, R)$ be a partial order, where $U = \{u_1, u_2, \ldots, u_n\}$. Furthermore let $V = \{v_1, v_2, \ldots, v_n\}$. The* domination bipartite graph $C(P) = (U, V, E)$ *is defined such that $u_iv_j \in E$ if and only if $u_i <_P u_j$.*

Lemma 1 ([9]). *Let $P = (U, R)$ be a partial order. Then, P is an interval order if and only if $C(P)$ is a chain graph.*

Extending the notion of $C(P)$, we now introduce the bipartite graph $NC(P)$ to characterize linear orders (cf. Lemma 2). Here "NC" stands for "Non-strictly Comparable". Namely, this graph can be obtained by adding to the graph $C(P)$ the perfect matching $\{u_iv_i \mid i = 1, 2, \ldots, n\}$ on the vertices of U and V.

Definition 8. *Let $P = (U, R)$ be a partial order, where $U = \{u_1, u_2, \ldots, u_n\}$. Furthermore let $V = \{v_1, v_2, \ldots, v_n\}$. Then, $NC(P) = (U, V, E)$ is the bipartite graph, such that $u_iv_j \in E$ if and only if $u_i \leq_P u_j$.*

Lemma 2. *Let $P = (U, R)$ be a partial order. Then, P is a linear order if and only if $NC(P)$ is a chain graph.*

Now we introduce the notion of a *linear-interval cover* of a bipartite graph. This notion is crucial for our main result of this section, cf. Theorem 3.

Definition 9. *Let $G = (U, V, E)$ be a bipartite graph, where $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_n\}$. Let $E_0 = \{u_iv_i \mid 1 \leq i \leq n\}$ and suppose that $E_0 \subseteq E$. Then, G is* linear-interval coverable *if there exist two chain graphs $G_1 = (U, V, E_1)$ and $G_2 = (U, V, E_2)$, such that $G = G_1 \cup G_2$ and $E_0 \subseteq E_2 \setminus E_1$. In this case, the sets $\{E_1, E_2\}$ are a* linear-interval cover *of G.*

Theorem 3. *Let $P = (U, R)$ be a partial order. In the bipartite complement $\widehat{C}(P)$ of the graph $C(P)$, denote $E_0 = \{u_i v_i \mid 1 \leq i \leq n\}$. The following three statements are equivalent:*

(a) $P = P_1 \cap P_2$, where P_1 is a linear order and P_2 is an interval order.
(b) $\widehat{C}(P) = \widehat{NC}(P_1) \cup \widehat{C}(P_2)$ for two partial orders P_1 and P_2 on V, where $\widehat{NC}(P_1)$ and $\widehat{C}(P_2)$ are chain graphs.
(c) $\widehat{C}(P)$ is linear-interval coverable, i.e. $\widehat{C}(P) = G_1 \cup G_2$ for two chain graphs $G_1 = (U, V, E_1)$ and $G_2 = (U, V, E_2)$, where $E_0 \subseteq E_2 \setminus E_1$.

Furthermore, a linear-interval cover of the bipartite graph $\widehat{C}(P)$ does not only guarantee that the input graph G is a PI graph, but it can be also used to efficiently compute a PI representation of G, as the next theorem states.

Theorem 4. *Let G be a cocomparability graph with n vertices and P be the partial order of $\overline{G}$. Let $\{E_1, E_2\}$ be a linear-interval cover of $\widehat{C}(P)$. Then we can construct in $O(n^2)$ time a PI representation R of G.*

5 Detecting Linear-Interval Covers Using Boolean Satisfiability

The natural algorithmic question that arizes from the characterization of PI graphs using linear-interval covers in Theorems 2 and 3, is the following: "Given a cocomparability graph G and a partial order P of $\overline{G}$, can we efficiently decide whether the bipartite graph $\widehat{C}(P)$ has a linear-interval cover?" We will answer this algorithmic question in the affirmative in Section 6. In this section we translate *every* instance of this decision problem (i.e. whether the bipartite graph $\widehat{C}(P)$ has a linear-interval cover) to a restricted instance of 3SAT (cf. Theorem 5). That is, for every such a bipartite graph $\widehat{C}(P)$, we construct a Boolean formula ϕ in conjunctive normal form (CNF), with size polynomial on the size of $\widehat{C}(P)$ (and thus also on G), such that $\widehat{C}(P)$ has a linear-interval cover if and only if ϕ is satisfiable. In particular, this formula ϕ can be written as $\phi = \phi_1 \wedge \phi_2$, where ϕ_1 has three literals in every clause and ϕ_2 has two literals in every clause. Moreover, as we will prove in Section 6, the satisfiability problem can be efficiently decided on the formula ϕ, by exploiting an appropriate sub-formula of ϕ which is gradually mixed (cf. Definition 2).

In the remainder of the paper, given a cocomparability graph G and a partial ordering P of its complement $\overline{G}$, we denote by $\widetilde{G} = \widehat{C}(P)$ the bipartite complement of the domination bipartite graph $C(P)$ of P. Furthermore we denote by H the associated split graph of $\widetilde{G}$ and by H^* the conflict graph of H. Moreover, we assume in the remainder of the paper without loss of generality that $\chi(H^*) \leq 2$, i.e. that H^* is bipartite. Indeed, as we can prove, if $\chi(H^*) > 2$ then $\widetilde{G}$ does not have a linear-interval cover, i.e. G is not a PI graph. Note that every proper 2-coloring of the vertices of the conflict graph H^* corresponds to exactly one 2-coloring of the edges of H that includes no monochromatic AC_4. We assume

in the following that a proper 2-coloring (with colors blue and red) of the vertices of H^* is given as input; note that χ_0 can be computed in polynomial time.

Let $C_1, C_2, \ldots, C_k$ be the connected components of H^*. Some of these components of H^* may be isolated vertices, which correspond to uncommitted edges in H. We assign to every component C_i, where $1 \leq i \leq k$, the Boolean variable x_i. Since H^* is bipartite by assumption, the vertices of each connected component C_i of H^* can be partitioned into two color classes $S_{i,1}$ and $S_{i,2}$. Without loss of generality, we assume that $S_{i,1}$ (resp. $S_{i,2}$) contains the vertices of C_i that are colored red (resp. blue) in χ_0. Note that, since vertices of H^* correspond to edges of H (cf. Definition 6), for every two edges e and e' of H that are in conflict (i.e. $e||e'$) there exists an index $i \in \{1, 2, \ldots, k\}$ such that one of these edges belongs to $S_{i,1}$ and the other belongs to $S_{i,2}$. We now assign a *literal* ℓ_e to every edge e of H as follows: if $e \in S_{i,1}$ for some $i \in \{1, 2, \ldots, k\}$, then $\ell_e = x_i$; otherwise, if $e \in S_{i,2}$, then $\ell_e = \overline{x_i}$. Note that, by construction, whenever two edges are in conflict in H, their assigned literals are one the negation of the other.

Observation 1. *Every truth assignment τ of the variables $x_1, x_2, \ldots, x_k$ corresponds bijectively to a proper 2-coloring χ_τ (with colors blue and red) of the vertices of H^*, as follows: $x_i = 0$ in τ (resp. $x_i = 1$ in τ), if and only if all vertices of the component C_i have in χ_τ the same color as in χ_0 (resp. opposite color than in χ_0). In particular, $\tau = (0, 0, \ldots, 0)$ corresponds to the coloring χ_0.*

Description of the 3-CNF Formula ϕ_1: Consider an AC_6 in the split graph H, and let e, e', e'' be its three edges in H, such that no two literals among $\{\ell_e, \ell_{e'}, \ell_{e''}\}$ are one the negation of the other. Then the Boolean formula ϕ_1 has for this triple $\{e, e', e''\}$ of edges exactly the two clauses $\alpha = (\ell_e \vee \ell_{e'} \vee \ell_{e''})$ and $\alpha' = (\overline{\ell_e} \vee \overline{\ell_{e'}} \vee \overline{\ell_{e''}})$. It is easy to check by the assignment of literals to edges that the clause α (resp. the clause α') of ϕ_1 is false in a truth assignment τ of the variables if and only if all edges $\{e, e', e''\}$ are colored red (resp. blue) in the 2-edge-coloring χ_τ of H (cf. Observation 1).

Consider now another AC_6 of H on the edges $\{e_1, e_2, e_3\}$, in which at least one literal among $\{\ell_{e_1}, \ell_{e_2}, \ell_{e_3}\}$ is the negation of another literal, for example $\ell_{e_1} = \overline{\ell_{e_2}}$. Then, for *any* proper 2-coloring of the vertices of H^*, the edges e and e' of H receive different colors, and thus this AC_6 is not monochromatic.

Observation 2. *The formula ϕ_1 is satisfied by a truth assignment τ if and only if the corresponding 2-coloring χ_τ of the edges of H does not contain any monochromatic AC_6.*

Description of the 2-CNF Formula ϕ_2: Denote for simplicity $H = (U, V, E_H)$, where $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_n\}$. Furthermore denote $E_0 = \{u_i v_i \mid 1 \leq i \leq n\}$. Let $E' = E_H \setminus E_0$ and $H' = H - E_0$, i.e. H' is the split graph that we obtain if we remove from H all edges of E_0. Consider now a pair of edges $e = u_i v_t$ and $e' = u_t v_j$ of E', such that $u_i v_j \notin E'$. Note that i and j may be equal. However, since $E' \cap E_0 = \emptyset$, it follows that $i \neq t$ and $t \neq j$.

Moreover, since the edge $u_t v_t$ belongs to E_H but not to E', it follows that the edges e and e' are in conflict in H' but not in H (for both cases where $i = j$ and $i \neq j$). That is, although e and e' are two non-adjacent vertices in the conflict graph H^* of H, they are adjacent vertices in the conflict graph of H'. For both cases where $i = j$ and $i \neq j$, an example of such a pair of edges $\{e, e'\}$ is illustrated in Figure 2. For every such pair $\{e, e'\}$ of edges in H, the Boolean formula ϕ_2 has the clause $(\ell_e \vee \ell_{e'})$. It is easy to check by the assignment of literals to edges of H that this clause $(\ell_e \vee \ell_{e'})$ of ϕ_2 is false in the truth assignment τ if and only if both e and e' are colored *red* in the 2-edge coloring χ_τ of H.

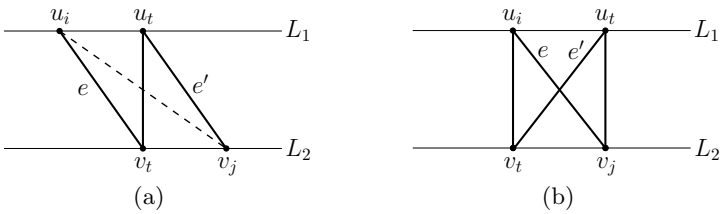

Fig. 2. Two edges $e = u_i v_t$ and $e' = u_t v_j$ of H, for which the formula ϕ_2 has the clause $(\ell_e \vee \ell_{e'})$, in the case where (a) $i \neq j$ and (b) $i = j$

Now we provide the main result of this section, which relates the existence of a linear-interval cover in $\widetilde{G} = \widehat{C}(P)$ with the satisfiability of the formula $\phi_1 \wedge \phi_2$.

Theorem 5. *$\widetilde{G} = \widehat{C}(P)$ is linear-interval colorable if and only if $\phi_1 \wedge \phi_2$ is satisfiable. Given a satisfying assignment τ of $\phi_1 \wedge \phi_2$, a linear-interval cover of $\widetilde{G}$ can be computed in $O(n^2)$ time.*

6 The Recognition of Linear-Interval Orders and PI Graphs

In this section we investigate the structure of the formula $\phi_1 \wedge \phi_2$ that we computed in Section 5. In particular, we first prove some fundamental structural properties of $\phi_1 \wedge \phi_2$, which allow us to find an appropriate sub-formula of $\phi_1 \wedge \phi_2$ which is gradually mixed (cf. Definition 2). Then we exploit this sub-formula of $\phi_1 \wedge \phi_2$ to provide an algorithm that solves the satisfiability problem on $\phi_1 \wedge \phi_2$ in time linear to its size, cf. Theorem 6. Finally, using this satisfiability algorithm, we combine our results of Sections 4 and 5 to recognize efficiently PI graphs and linear-interval orders.

The main structural properties of $\phi_1 \wedge \phi_2$ are proved in Lemmas 3 and 4. The proof of the next lemma is a based on the results of [13].

Lemma 3. *Let α and β be two clauses of ϕ_1. If α and β share at least one variable, then $\{\alpha, \overline{\alpha}\} = \{\beta, \overline{\beta}\}$.*

Algorithm 1. Recognition of PI graphs

Input: A graph $G = (V, E)$
Output: A PI representation R of G, or the announcement that G is not a PI graph

if G is a trapezoid graph **then**
  Compute a partial order P of the complement $\overline{G}$
else return "G is not a PI graph"
Compute the domination bipartite graph $C(P)$ from P
$\widetilde{G} \leftarrow \widehat{C}(P)$
Compute the associated split graph H of $\widetilde{G}$
Compute the conflict graph H^* of H
if H^* is bipartite **then**
  Compute a 2-coloring χ_0 of the vertices of H^*
  Compute the formulas ϕ_1 and ϕ_2
  if $\phi_1 \wedge \phi_2$ is satisfiable **then**
    Compute a satisfying truth assignment τ of $\phi_1 \wedge \phi_2$ by Theorem 6
    Compute from τ a linear-order cover of $\widetilde{G}$ by Theorem 5
    Compute a PI representation R of G by Theorem 4
  else
    return "G is not a PI graph"
else
  return "G is not a PI graph"
return R

Definition 10. *The clauses of ϕ_2 are partitioned into the* sub-formulas ϕ_2', ϕ_2'', *such that ϕ_2' contains all tautologies of ϕ_2 and all clauses of ϕ_2 in which at least one literal corresponds to an uncommitted edge, while ϕ_2'' contains all the remaining clauses of ϕ_2.*

Lemma 4. *Let $\{e_1, e_2, e_3\}$ be the three edges of an AC_6 in H, which has clauses in ϕ_1. Let e be an edge of H such that $(\ell_e \vee \ell_{e_1})$ is a clause in ϕ_2''. Then ϕ_2'' has also one of the clauses $\{(\ell_e \vee \overline{\ell_{e_2}}), (\ell_e \vee \overline{\ell_{e_3}})\}$.*

The next corollary, which follows easily by Definition 2 and by Lemmas 3 and 4, allows us to use the linear time algorithm for gradually mixed formulas (cf. Theorem 1) in order to solve the satisfiability problem on $\phi_1 \wedge \phi_2''$.

Corollary 1. *$\phi_1 \wedge \phi_2''$ is a gradually mixed formula.*

In the next theorem we use Corollary 1 to design an algorithm that decides satisfiability on $\phi_1 \wedge \phi_2$ in time linear to its size. This will enable us to combine the results of Sections 4 and 5 to recognize efficiently whether a given graph is a PI graph, or equivalently, due to Theorem 2, whether a given partial order P is the intersection of a linear order P_1 and an interval order P_2.

Theorem 6. *$\phi_1 \wedge \phi_2$ is satisfiable if and only if $\phi_1 \wedge \phi_2''$ is satisfiable. Given a satisfying truth assignment of $\phi_1 \wedge \phi_2''$ we can compute a satisfying truth assignment of $\phi_1 \wedge \phi_2$ in linear time.*

Now we are ready to present our recognition algorithm for PI graphs (Algorithm 1). Its correctness and timing analysis is established in Theorem 7. Due to characterization of PI graphs in Theorem 2 using partial orders, Theorem 8 follows also by Theorem 7.

Theorem 7. *Let $G = (V, E)$ be a graph and $\overline{G} = (V, \overline{E})$ be its complement, where $|V| = n$ and $|\overline{E}| = m$. Then Algorithm 1 constructs in $O(n^2 m)$ time a PI representation of G, or it announces that G is not a PI graph.*

Theorem 8. *Let $P = (U, R)$ be a partial order, where $|U| = n$ and $|R| = m$. Then we can decide in $O(n^2 m)$ time whether P is a linear-interval order, and in this case we can compute a linear order P_1 and an interval order P_2 such that $P = P_1 \cap P_2$.*

References

1. Benzer, S.: On the topology of the genetic fine structure. Proc. of the National Academy of Sciences (PNAS) 45, 1607–1620 (1959)
2. Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph classes: a survey. In: Society for Industrial and Applied Mathematics (SIAM) (1999)
3. Carrano, A.V.: Establishing the order to human chromosome-specific DNA fragments. In: Biotechnology and the Human Genome, pp. 37–50. Plenum Press (1988)
4. Corneil, D.G., Kamula, P.A.: Extensions of permutation and interval graphs. In: Proceedings of the 18th Southeastern Conference on Combinatorics, Graph Theory and Computing, pp. 267–275 (1987)
5. Dagan, I., Golumbic, M.C., Pinter, R.Y.: Trapezoid graphs and their coloring. Discrete Applied Mathematics 21(1), 35–46 (1988)
6. Goldberg, P.W., Golumbic, M.C., Kaplan, H., Shamir, R.: Four strikes against physical mapping of DNA. Journal of Computational Biology 2(1), 139–152 (1995)
7. Golumbic, M.C.: Algorithmic graph theory and perfect graphs, 2nd edn. Annals of Discrete Mathematics, vol. 57. North-Holland Publishing Co. (2004)
8. Kaufmann, M., Kratochvil, J., Lehmann, K.A., Subramanian, A.R.: Max-tolerance graphs as intersection graphs: cliques, cycles, and recognition. In: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 832–841 (2006)
9. Ma, T.-H., Spinrad, J.P.: On the 2-chain subgraph cover and related problems. Journal of Algorithms 17, 251–268 (1994)
10. Mahadev, N., Peled, U.N.: Threshold Graphs and Related Topics. Annals of Discrete Mathematics, vol. 56. North-Holland Publishing Co. (1995)
11. Mertzios, G.B.: The recognition of triangle graphs. Theoretical Computer Science 438, 34–47 (2012)
12. Mertzios, G.B., Sau, I., Zaks, S.: The recognition of tolerance and bounded tolerance graphs. SIAM Journal on Computing 40(5), 1234–1257 (2011)
13. Raschle, T., Simon, K.: Recognition of graphs with threshold dimension two. In: Proceedings of the 27th ACM Symposium on Theory of Computing (STOC), pp. 650–661 (1995)
14. Spinrad, J.P.: Efficient graph representations. Fields Institute Monographs, vol. 19. American Mathematical Society (2003)
15. Yannakakis, M.: The complexity of the partial order dimension problem. SIAM Journal on Algebraic and Discrete Methods 3, 351–358 (1982)