RWTH Aachen

Department of Computer Science
Technical Report

The Recognition of Tolerance and Bounded Tolerance Graphs is NP-complete

George B. Mertzios
Ignasi Sau
Shmuel Zaks

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2009-06

RWTH Aachen · Department of Computer Science · April 2009

The publications of the Department of Computer Science of *RWTH Aachen University* are in general accessible through the World Wide Web.

http://aib.informatik.rwth-aachen.de/

The Recognition of Tolerance and Bounded Tolerance Graphs is NP-complete

George B. Mertzios*, Ignasi Sau†, and Shmuel Zaks‡

Abstract. Tolerance graphs model interval relations in such a way that intervals can tolerate a certain degree of overlap without being in conflict. This class of graphs has been extensively studied, due to both its interesting structure and its numerous applications (in bioinformatics, constrained-based temporal reasoning, resource allocation, and scheduling problems, among others). Several efficient algorithms for optimization problems that are NP-hard in general graphs have been designed for tolerance graphs. In spite of this, the recognition of tolerance graphs -namely, the problem of deciding whether a given graph is a tolerance graph- as well as the recognition of their main subclass of bounded tolerance graphs, are probably the most fundamental open problems in this context (cf. the book on tolerance graphs [14]) since their introduction almost three decades ago [11]. In this article we resolve this problem, by proving that both recognition problems are NP-complete, even in the case where the input graph is a trapezoid graph. For our reduction we extend the notion of an acyclic orientation of permutation and trapezoid graphs. Our main tool is a new algorithm (which uses an approach similar to [6]) that transforms a given trapezoid graph into a permutation graph, while preserving this new acyclic orientation property.

Keywords: Tolerance graphs, bounded tolerance graphs, recognition, NP-complete, trapezoid graphs, permutation graphs.

1 Introduction

1.1 Tolerance graphs and related graph classes

A simple undirected graph $G = (V, E)$ on n vertices is a *tolerance* graph if there exists a collection $I = \{I_i \mid i = 1, 2, \ldots, n\}$ of closed intervals on the real line and a set $t = \{t_i \mid i = 1, 2, \ldots, n\}$ of positive numbers, such that for any two vertices $v_i, v_j \in V$, $v_i v_j \in E$ if and only if $|I_i \cap I_j| \geq \min\{t_i, t_j\}$. The pair $\langle I, t\rangle$ is called a *tolerance representation* of G. If G has a tolerance representation $\langle I, t\rangle$, such that $t_i \leq |I_i|$ for every $i = 1, 2, \ldots, n$, then G is called a *bounded tolerance* graph and $\langle I, t\rangle$ a *bounded tolerance representation* of G.

Tolerance graphs were introduced in [11], in order to generalize some of the well known applications of interval graphs. The main motivation was in the context of resource allocation and scheduling problems, in which resources, such as rooms and vehicles, can tolerate sharing among users [14]. If we replace in the definition of tolerance graphs the operator *min* by the operator *max*, we obtain the class of *max-tolerance* graphs. Both tolerance and max-tolerance graphs find in a natural way applications in biology and bioinformatics, as in the comparison of DNA sequences from different organisms or individuals [18], by making use of a software tool like BLAST [1]. Tolerance graphs find numerous other applications in constrained-based temporal reasoning, data transmission through networks to efficiently scheduling aircraft and crews, as well as contributing to genetic analysis and studies of the brain [13, 14]. This class of graphs has attracted many research efforts [2, 4, 8, 12–14, 16, 19, 23, 24], as it generalizes in a

*Department of Computer Science, RWTH Aachen University, Germany. Email: mertzios@cs.rwth-aachen.de

†Mascotte joint Project of INRIA/CNRS/UNSA, Sophia-Antipolis, France; and Graph Theory and Combinatorics Group, Applied Maths. IV Dept. of UPC, Barcelona, Spain. Email: ignasi.sau@sophia.inria.fr

‡Department of Computer Science, Technion, Haifa, Israel. Email: zaks@cs.technion.ac.il

natural way both interval graphs (when all tolerances are equal) and permutation graphs (when $t_i = |I_i|$ for every $i = 1, 2, \ldots, n$) [11]. For a detailed survey on tolerance graphs we refer to [14].

A graph is *perfect* if the chromatic number of every induced subgraph equals the clique number of that subgraph. Several difficult combinatorial problems can be solved efficiently, i.e. in polynomial time, on the class of perfect graphs, such as minimum coloring, maximum clique, and independent set [15]. Thus, since the class of tolerance graphs is a subclass of perfect graphs [12], there exist polynomial algorithms for these problems on tolerance and bounded tolerance graphs as well. In spite of this, faster algorithms have been designed for tolerance and bounded tolerance graphs, which exploit their special structure [13, 14, 23, 24].

A *comparability* graph is a graph which can be transitively oriented. A *co-comparability* graph is a graph whose complement is a comparability graph. A *trapezoid* (resp. *parallelogram* and *permutation*) graph is the intersection graph of trapezoids (resp. parallelograms and line segments) between two parallel lines L_1 and L_2 [10]. Such a representation with trapezoids (resp. parallelograms and line segments) is called a *trapezoid* (resp. *parallelogram* and *permutation*) *representation* of this graph. A graph is bounded tolerance if and only if it is a parallelogram graph [2]. Permutation graphs are a strict subset of parallelogram graphs [3]. Furthermore, parallelogram graphs are a strict subset of trapezoid graphs [26], and both are subsets of co-comparability graphs [10, 14]. On the contrary, tolerance graphs are not even co-comparability graphs [10, 14]. Recently, we have presented in [23] a natural intersection model for general tolerance graphs, given by parallelepipeds in the three-dimensional space. This representation generalizes the parallelogram representation of bounded tolerance graphs, and has been used to improve the time complexity of minimum coloring, maximum clique, and weighted independent set on tolerance graphs [23].

Although tolerance and bounded tolerance graphs have been studied extensively, the recognition problems for both these classes are probably the most fundamental open problems since their introduction [5,10,14]. Therefore, all existing algorithms assume that, along with the input tolerance graph, a tolerance representation of it is given. The only result about the complexity of recognizing tolerance and bounded tolerance graphs is that they have a polynomial sized tolerance representation, hence the problems of tolerance and bounded tolerance graph recognition are in the class NP [16]. However, the recognition of max-tolerance graphs is known to be NP-hard [18]. On the contrary, a linear time recognition algorithm of *bipartite tolerance* graphs has been recently presented [5]. Furthermore, the class of *bounded bitolerance* graphs, which is equivalent to that of trapezoid graphs [20], can be also recognized in polynomial time [6, 21, 22].

1.2 Our contribution

In this article, we resolve the problems of recognizing both tolerance and bounded tolerance graphs. In particular, we prove that both problems are NP-complete, by providing a reduction from the monotone-Not-All-Equal-3-SAT (monotone-NAE-3-SAT) problem. Consider a boolean formula ϕ in conjunctive normal form with three literals in every clause (3-CNF), which is monotone, i.e. no variable is negated. The formula ϕ is called NAE-satisfiable if there exists a truth assignment of the variables of ϕ, such that every clause has at least one true variable and one false variable. Given a monotone 3-CNF formula ϕ, we construct a trapezoid graph H_ϕ, which is parallelogram, i.e. bounded tolerance, if and only if ϕ is NAE-satisfiable. Moreover, we prove that the constructed graph H_ϕ is tolerance if and only if it is bounded tolerance. Thus, since the recognition of tolerance and of bounded tolerance graphs are in the class NP [16], it follows that these problems are both NP-complete. Actually, our results imply that the recognition problems remain NP-complete even if the given graph is trapezoid, since the constructed graph H_ϕ is trapezoid.

For our reduction we extend the notion of an acyclic orientation of permutation and trapezoid graphs. Our main tool is a new algorithm (which uses an approach similar to [6]) that transforms a given trapezoid graph into a permutation graph, while preserving this new acyclic orientation property. The constructed permutation graph does not depend on any particular trapezoid representation of the input graph G, and this is one of the main advantages of this algorithm.

Organization of the paper. We first present in Section 2 several properties of permutation and trapezoid graphs, as well as the algorithm Split-U, which constructs a permutation graph from a trapezoid graph. In Section 3 we present the reduction of the monotone-NAE-3-SAT problem to the recognition of bounded tolerance graphs. In Section 4 we prove that this reduction can be extended to the recognition of general tolerance graphs. Finally, we discuss the presented results and further research directions in Section 5.

2 Trapezoid graphs and representations

In this section we first introduce (in Section 2.1) the notion of an *acyclic representation* of permutation and of trapezoid graphs. This is followed (in Section 2.2) by some structural properties of trapezoid graphs, which will be used in the sequel for the splitting algorithm Split-U. Given a trapezoid graph G and a vertex subset U of G with certain properties, this algorithm constructs a permutation graph $G^{\#}(U)$ with $2|U|$ vertices, which is independent on any particular trapezoid representation of the input graph G.

Notation. We consider in this article simple undirected and directed graphs with no loops or multiple edges. In an undirected graph G, the edge between vertices u and v is denoted by uv, and in this case u and v are said to be *adjacent* in G. If the graph G is directed, we denote by uv the arc from u to v. Given a graph $G = (V, E)$ and a subset $S \subseteq V$, $G[S]$ denotes the induced subgraph of G on the vertices in S, and we use $E[S]$ to denote $E(G[S])$. Whenever we deal with a trapezoid (resp. permutation and bounded tolerance, i.e. parallelogram) graph, we will consider w.l.o.g. a trapezoid (resp. permutation and parallelogram) representation, in which all endpoints of the trapezoids (resp. line segments and parallelograms) are distinct [9,14,17]. Given a permutation graph P along with a permutation representation R, we may not distinguish in the following between a vertex of P and the corresponding line segment in R, whenever it is clear from the context. Furthermore, with a slight abuse of notation, we will refer to the line segments of a permutation representation just as *lines*.

2.1 Acyclic permutation and trapezoid representations

Let $P = (V, E)$ be a permutation graph and R be a permutation representation of P. For a vertex $u \in V$, denote by $\theta_R(u)$ the angle of the line of u with L_2 in R. The class of permutation graphs is the intersection of comparability and co-comparability graphs [10]. Thus, given a permutation representation R of P, we can define two partial orders $(V, <_R)$ and $(V, \ll_R)$ on the vertices of P [10]. Namely, for two vertices u and v of G, $u <_R v$ if and only if $uv \in E$ and $\theta_R(u) < \theta_R(v)$, while $u \ll_R v$ if and only if $uv \notin E$ and u lies to the left of v in R. The partial order $(V, <_R)$ implies a transitive orientation Φ_R of P, such that $uv \in \Phi_R$ whenever $u <_R v$.

Let $G = (V, E)$ be a trapezoid graph, and R be a trapezoid representation of G, where for any vertex $u \in V$, the trapezoid corresponding to u in R is denoted by T_u. Since trapezoid graphs are also co-comparability graphs [10], we can similarly define the partial order $(V, \ll_R)$ on the vertices of G, such that $u \ll_R v$ if and only if $uv \notin E$ and T_u lies completely to the left of T_v in R. In this case, we may denote also $T_u \ll_R T_v$, instead of $u \ll_R v$.

In a given trapezoid representation R of a trapezoid graph G, we denote by $l(T_u)$ and $r(T_u)$ the left and the right line of T_u in R, respectively. Similarly to the case of permutation graphs, we use the relation $\ll_R$ for the lines $l(T_u)$ and $r(T_u)$, e.g. $l(T_u) \ll_R r(T_v)$ means that the line $l(T_u)$ lies to the left of the line $r(T_v)$ in R. Moreover, if the trapezoids of all vertices of a subset $S \subseteq V$ lie completely to the left (resp. right) of the trapezoid T_u in R, we write $R(S) \ll_R T_u$ (resp. $T_u \ll_R R(S)$). Note that there are several trapezoid representations of a particular trapezoid graph G. Given one such representation R, we can obtain another one R' by *vertical axis flipping* of R, i.e. R' is the mirror image of R along an imaginary line perpendicular to L_1 and L_2. Moreover, we can obtain another representation R'' of G by *horizontal axis flipping* of R, i.e. R'' is the mirror image of R along an imaginary line parallel to L_1 and L_2. We will use extensively these two basic operations throughout the article.

Definition 1. *Let P be a permutation graph with $2n$ vertices $\{u_1^1, u_1^2, u_2^1, u_2^2, \ldots, u_n^1, u_n^2\}$. Let R be a permutation representation and Φ_R be the corresponding transitive orientation of P. The simple directed graph F_R is obtained by merging u_i^1 and u_i^2 into a single vertex u_i, for every $i = 1, 2, \ldots, n$, where the arc directions of F_R are implied by the corresponding directions in Φ_R. Then,*

1. *R is an* acyclic permutation representation with respect to $\{u_i^1, u_i^2\}_{i=1}^n$*, *if F_R has no directed cycle,*
2. *P is an* acyclic permutation graph with respect to $\{u_i^1, u_i^2\}_{i=1}^n$, *if P has an acyclic representation R with respect to $\{u_i^1, u_i^2\}_{i=1}^n$.*

In Figure 1 we show an example of a permutation graph P with six vertices in Figure 1(a), a permutation representation R of P in Figure 1(b), the transitive orientation Φ_R of P in Figure 1(c), and the corresponding simple directed graph F_R in Figure 1(d). In the figure, the pairs $\{u_i^1, u_i^2\}_{i=1}^3$ are grouped inside ellipses. In this example, R is not an acyclic permutation representation with respect to $\{u_i^1, u_i^2\}_{i=1}^3$, since F_R has a directed cycle of length two. However, note that, by exchanging the lines u_1^1 and u_2^1 in R, the resulting permutation representation R' is acyclic with respect to $\{u_i^1, u_i^2\}_{i=1}^3$, and thus P is acyclic with respect to $\{u_i^1, u_i^2\}_{i=1}^3$.

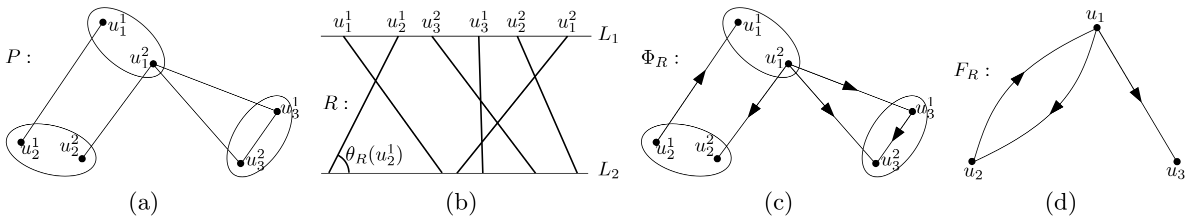

Fig. 1. (a) A permutation graph P, (b) a permutation representation R of P, (c) the transitive orientation Φ_R of P, and (d) the corresponding simple directed graph F_R.

Definition 2. *Let G be a trapezoid graph with n vertices and R be a trapezoid representation of G. Let P be the permutation graph with $2n$ vertices corresponding to the left and right lines of the trapezoids in R, R_P be the permutation representation of P induced by R, and $\{u_i^1, u_i^2\}$ be the vertices of P that correspond to the same vertex u_i of G, $i = 1, 2, \ldots, n$. Then,*

1. *R is an* acyclic trapezoid representation, *if R_P is an acyclic permutation representation with respect to $\{u_i^1, u_i^2\}_{i=1}^n$,*
2. *G is an* acyclic trapezoid graph, *if it has an acyclic representation R.*

*To simplify the presentation, we use throughout the paper $\{u_i^1, u_i^2\}_{i=1}^n$ to denote the set of n unordered pairs $\{u_1^1, u_1^2\}, \{u_2^1, u_2^2\}, \ldots, \{u_n^1, u_n^2\}$.

The following lemma follows easily from Definitions 1 and 2.

Lemma 1. *Any parallelogram graph is an acyclic trapezoid graph.*

Proof. Let G be a parallelogram graph with n vertices $\{u_1, u_2, \ldots, u_n\}$ and R be a parallelogram representation of G. That is, R is a trapezoid representation of G, such that the left and right lines $l(T_{u_i})$ and $r(T_{u_i})$ of the trapezoid T_{u_i}, $i = 1, 2, \ldots, n$, are parallel in R, i.e. $\theta_R(l(T_{u_i})) = \theta_R(r(T_{u_i}))$. Let P be the permutation graph with $2n$ vertices $\{u_1^1, u_1^2, u_2^1, u_2^2, \ldots, u_n^1, u_n^2\}$ corresponding to the left and right lines of the trapezoids of G in R, i.e. the vertices u_i^1 and u_i^2 correspond to $l(T_{u_i})$ and $r(T_{u_i})$, $i = 1, 2, \ldots, n$, respectively. Let R_P be the permutation representation of P induced by R, and Φ_{R_P} be the corresponding transitive orientation of the permutation graph P. Recall that, for two intersecting lines a, b in R_P, it holds $ab \in \Phi_{R_P}$ whenever $\theta_R(a) < \theta_R(b)$. It follows that for any $i = 1, 2, \ldots, n$, the pair $\{u_i^1, u_i^2\}$ of vertices in P has incoming edges from (resp. outgoing edges to) vertices of other pairs $\{u_j^1, u_j^2\}$ in Φ_{R_P}, which have smaller (resp. greater) angle with the line L_2 in R_P. Thus, the simple directed graph F_{R_P} defined in Definition 1 has no directed cycles, and therefore R_P is an acyclic permutation representation with respect to $\{u_i^1, u_i^2\}_{i=1}^n$, i.e. R is an acyclic trapezoid representation of G by Definition 2. □

2.2 Structural properties of trapezoid graphs

In the following, we state some definitions concerning an arbitrary simple undirected graph $G = (V, E)$, which are useful for our analysis. Although these definitions apply to any graph, we will use them only for trapezoid graphs. Similar definitions, for the restricted case where the graph G is connected, were studied in [6]. For $u \in V$ and $U \subseteq V$, $N(u) = \{v \in V \mid uv \in E\}$ is the set of adjacent vertices of u in G, $N[u] = N(u) \cup \{u\}$, and $N(U) = \bigcup_{u \in U} N(u) \setminus U$. If $N(U) \subseteq N(W)$ for two vertex subsets U and W, then U is said to be *neighborhood dominated* by W. Clearly, the relationship of neighborhood domination is transitive.

Let $C_1, C_2, \ldots, C_\omega$, $\omega \geq 1$, be the connected components of $G \setminus N[u]$ and $V_i = V(C_i)$, $i = 1, 2, \ldots, \omega$. For simplicity of the presentation, we will identify in the sequel the component C_i and its vertex set V_i, $i = 1, 2, \ldots, \omega$. For $i = 1, 2, \ldots, \omega$, the *neighborhood domination closure* of V_i with respect to u is the set $D_u(V_i) = \{V_p \mid N(V_p) \subseteq N(V_i),\ p = 1, 2, \ldots, \omega\}$ of connected components of $G \setminus N[u]$. A component V_i is called a *master component* of u if $|D_u(V_i)| \geq |D_u(V_j)|$ for all $j = 1, 2, \ldots, \omega$. The *closure complement* of the neighborhood domination closure $D_u(V_i)$ is the set $D_u^*(V_i) = \{V_1, V_2, \ldots, V_\omega\} \setminus D_u(V_i)$. Finally, for a subset $S \subseteq \{V_1, V_2, \ldots, V_\omega\}$, a component V_j of S is called *maximal*, if there is no component $V_k \in S$, such that $N(V_j) \subsetneqq N(V_k)$.

For example, consider the trapezoid graph G with vertex set $\{u, u_1, u_2, u_3, v_1, v_2, v_3, v_4\}$, which is given by the trapezoid representation R of Figure 2. The connected components of $G \setminus N[u] = \{v_1, v_2, v_3, v_4\}$ are $V_1 = \{v_1\}$, $V_2 = \{v_2\}$, $V_3 = \{v_3\}$, and $V_4 = \{v_4\}$. Then, $N(V_1) = \{u_1\}$, $N(V_2) = \{u_1, u_3\}$, $N(V_3) = \{u_2, u_3\}$, and $N(V_4) = \{u_3\}$. Hence, $D_u(V_1) = \{V_1\}$, $D_u(V_2) = \{V_1, V_2, V_4\}$, $D_u(V_3) = \{V_3, V_4\}$, and $D_u(V_4) = \{V_4\}$; thus, V_2 is the only master component of u. Furthermore, $D_u^*(V_1) = \{V_2, V_3, V_4\}$, $D_u^*(V_2) = \{V_3\}$, $D_u^*(V_3) = \{V_1, V_2\}$ and $D_u^*(V_4) = \{V_1, V_2, V_3\}$.

Lemma 2. *Let G be a simple graph, u be a vertex of G, and let $V_1, V_2, \ldots, V_\omega$, $\omega \geq 1$, be the connected components of $G \setminus N[u]$. If V_i is a master component of u, such that $D_u^*(V_i) \neq \emptyset$, then $D_u^*(V_j) \neq \emptyset$ for every component V_j of $G \setminus N[u]$.*

Proof. Since V_i is a master component, and since $D_u^*(V_i) \neq \emptyset$, it follows that $|D_u(V_j)| \leq |D_u(V_i)| < \omega$, for every connected component $V_j \in \{V_1, V_2, \ldots, V_\omega\}$. Therefore, $|D_u(V_j)| < \omega$, and thus, $D_u^*(V_j) \neq \emptyset$ as well. □

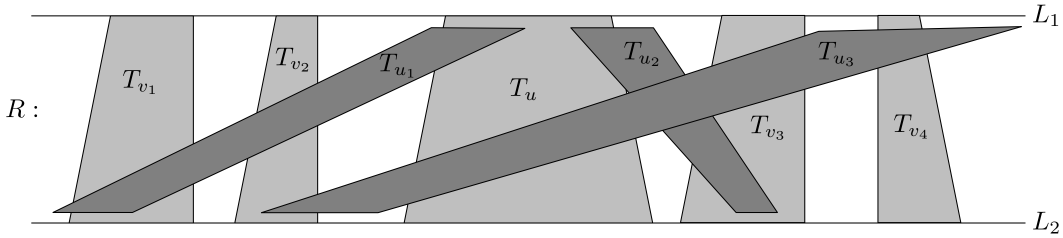

Fig. 2. A trapezoid representation R of a trapezoid graph G.

In the following we investigate several properties of trapezoid graphs, in order to derive the vertex-splitting algorithm Split-U in Section 2.3.

Remark 1. Similar properties of trapezoid graphs have been studied in [6], leading to another vertex-splitting algorithm, called Split-All. However, the algorithm proposed in [6] is incorrect, since it is based on an incorrect property†, as was also verified by [7]. In the sequel of this section, we present new definitions and properties. In the cases where a silimarity arises with those of [6], we refer to it specifically.

The following lemma, which has been stated in Observation 3.1(4) in [6] (withought a proof), will be used in our analysis below. For the sake of completeness, we present here the proof.

Lemma 3. *Let R be a trapezoid representation of a trapezoid graph G, and V_i be a master component of a vertex u of G, such that $R(V_i) \ll_R T_u$. Then, $T_u \ll_R R(V_j)$ for every component $V_j \in D_u^*(V_i)$.*

Proof. Suppose otherwise that $R(V_j) \ll_R T_u$, for some $V_j \in D_u^*(V_i)$. Consider first the case where $R(V_j) \ll_R R(V_i) \ll_R T_u$. Then, since V_i lies between V_j and T_u in R, all trapezoids that intersect T_u and V_j, must intersect also V_i. Thus, $N(V_j) \subseteq N(V_i)$, i.e. $V_j \in D_u(V_i)$, which is a contradiction, since $V_j \in D_u^*(V_i)$. Consider now the case where $R(V_i) \ll_R R(V_j) \ll_R T_u$. Then, we obtain similarly that $N(V_i) \subseteq N(V_j)$, and thus, $D_u(V_i) \subseteq D_u(V_j)$. Since $V_j \in D_u(V_j) \setminus D_u(V_i)$, it follows that $|D_u(V_i)| < |D_u(V_j)|$. This is a contradiction to the assumption that V_i is a master component of u. Thus, $T_u \ll_R R(V_j)$ for any $V_j \in D_u^*(V_i)$. □

In the following two definitions, we partition the neighbors of a vertex in a trapezoid graph G into four possibly empty sets. In the first definition, these sets depend only on the graph G itself, while in the second one, they depend on a particular trapezoid representation of G.

Definition 3. *Let G be a trapezoid graph, and u be a vertex of G. Let V_i be a master component of u, such that $D_u^*(V_i) \neq \emptyset$, and V_j be a maximal component of $D_u^*(V_i)$. Then, the vertices of $N(u)$ are partitioned into four possibly empty sets:*

1. *$N_0(u, V_i, V_j)$: vertices not adjacent to either V_i or V_j.*
2. *$N_1(u, V_i, V_j)$: vertices adjacent to V_i but not to V_j.*
3. *$N_2(u, V_i, V_j)$: vertices adjacent to V_j but not to V_i.*
4. *$N_{12}(u, V_i, V_j)$: vertices adjacent to both V_i and V_j.*

†In Observation 3.1(5) of [6], it is claimed that for an arbitrary trapezoid representation R of a connected trapezoid graph G, where V_i is a master component of u, such that $D_u^*(V_i) \neq \emptyset$ and $R(V_i) \ll_R T_u$, it holds $R(D_u(V_i)) \ll_R T_u \ll_R R(D_u^*(V_i))$. However, the first part of the latter inequality is not true. For instance, in the trapezoid graph G of Figure 2, V_2 is a master component of u, where $D_u^*(V_2) = \{V_3\} \neq \emptyset$ and $R(V_2) \ll_R T_u$. However, $V_4 = \{v_4\} \in D_u(V_2)$ and $T_u \ll_R T_{v_4}$, and thus, $R(D_u(V_2)) \not\ll_R T_u$.

Definition 4. *Let G be a trapezoid graph, R be a representation of G, and u be a vertex of G. Denote by $D_1(u,R)$ and $D_2(u,R)$ the sets of trapezoids of R that lie completely to the left and to the right of T_u in R, respectively. Then, the vertices of $N(u)$ are partitioned into four possibly empty sets:*

1. *$N_0(u,R)$: vertices not adjacent to either $D_1(u,R)$ or $D_2(u,R)$.*
2. *$N_1(u,R)$: vertices adjacent to $D_1(u,R)$ but not to $D_2(u,R)$.*
3. *$N_2(u,R)$: vertices adjacent to $D_2(u,R)$ but not to $D_1(u,R)$.*
4. *$N_{12}(u,R)$: vertices adjacent to both $D_1(u,R)$ and $D_2(u,R)$.*

The following lemma connects Definitions 3 and 4. The intuition behind this lemma is that the sets defined in Definition 3 include those neighbors of u, whose trapezoids intersect some trapezoids that lie to the left and/or to the right of T_u in a trapezoid representation of G.

Lemma 4. *Let G be a trapezoid graph, R be a representation of G, and u be a vertex of G. Let V_i be a master component of u, such that $D_u^*(V_i) \neq \emptyset$, and let V_j be a maximal component of $D_u^*(V_i)$. If $R(V_i) \ll_R T_u$, then $N_X(u,V_i,V_j) = N_X(u,R)$ for every $X \in \{0,1,2,12\}$.*

Proof. Since $D_u^*(V_i) \neq \emptyset$ and $R(V_i) \ll_R T_u$, it follows by Lemma 3 that $T_u \ll_R R(V_j)$, i.e. $V_j \in D_2(u,R)$. Suppose that a component $V_\ell \neq V_j$ is the leftmost one of $D_2(u,R)$ in R, i.e. $T_u \ll_R R(V_\ell) \ll_R R(V_j)$. Since V_ℓ lies between T_u and V_j in R, all trapezoids that intersect T_u and V_j, must also intersect V_ℓ, and thus, $N(V_j) \subseteq N(V_\ell)$. It follows that $V_\ell \in D_u^*(V_i)$, i.e. $V_\ell \notin D_u(V_i)$, since otherwise $V_j \in D_u(V_i)$, which is a contradiction. Furthermore, since V_j is a maximal component of $D_u^*(V_i)$, and since $N(V_j) \subseteq N(V_\ell)$, it follows that $N(V_j) = N(V_\ell)$, i.e. $N_X(u,V_i,V_j) = N_X(u,V_i,V_\ell)$ for every $X \in \{0,1,2,12\}$.

Suppose that a component $V_k \neq V_i$ is the rightmost one of $D_1(u,R)$ in R, i.e. $R(V_i) \ll_R R(V_k) \ll_R T_u$. Then, $V_k \in D_u(V_i)$, since otherwise $T_u \ll_R R(V_k)$ by Lemma 3, which is a contradiction. Thus, $N(V_k) \subseteq N(V_i)$. Furthermore, since V_k lies between V_j and T_u in R, all trapezoids that intersect T_u and V_j, must also intersect V_k, and thus, $N(V_i) \subseteq N(V_k)$. Therefore, $N(V_i) = N(V_k)$, i.e. $N_X(u,V_i,V_\ell) = N_X(u,V_k,V_\ell)$ for every $X \in \{0,1,2,12\}$, and thus, $N_X(u,V_i,V_j) = N_X(u,V_k,V_\ell)$ for every $X \in \{0,1,2,12\}$.

Consider now a vertex $v \in N(u)$, and recall that V_k (resp. V_ℓ) is the rightmost (resp. leftmost) component of $D_1(u,R)$ (resp. $D_2(u,R)$) in R. Thus, if T_v intersects at least one component of $D_1(u,R)$ (resp. $D_2(u,R)$), then T_v intersects also with V_k (resp. V_ℓ). On the other hand, if T_v does not intersect any component of $D_1(u,R)$ (resp. $D_2(u,R)$), then T_v clearly does not intersect V_k (resp. V_ℓ), since $V_k \subseteq D_1(u,R)$ (resp. $V_j \subseteq D_2(u,R)$). It follows that $N_X(u,V_k,V_\ell) = N_X(u,R)$, and thus, $N_X(u,V_i,V_j) = N_X(u,R)$ for every $X \in \{0,1,2,12\}$. This proves the lemma. $\square$

Note that, given a trapezoid representation R of G, we may assume in Lemma 4 w.l.o.g. that $R(V_i) \ll_R T_u$, by possibly performing a vertical axis flipping of R. Thus, we can state now the following definition of the sets D_u and D_u^*, regardless of the choice the components V_i and V_j of u.

Definition 5. *Let G be a trapezoid graph, u be a vertex of G, and V_i be an arbitrarily chosen master component of u. Then, $\delta_u = V_i$ and*

1. *if $D_u^*(V_i) = \emptyset$, then $\delta_u^* = \emptyset$.*
2. *if $D_u^*(V_i) \neq \emptyset$, then $\delta_u^* = V_j$, for an arbitrarily chosen maximal component $V_j \in D_u^*(V_i)$.*

Definition 6. *Let G be a trapezoid graph and u be a vertex of G. The vertices of $N(u)$ are partitioned into four possibly empty sets:*

1. $N_0(u)$: *vertices not adjacent to either* δ_u *or* δ_u^*.
2. $N_1(u)$: *vertices adjacent to* δ_u *but not to* δ_u^*.
3. $N_2(u)$: *vertices adjacent to* δ_u^* *but not to* δ_u.
4. $N_{12}(u)$: *vertices adjacent to both* δ_u *and* δ_u^*.

Suppose that $\delta_u^* \neq \emptyset$, and let V_i be the master component of u and V_j be the maximal component of $D_u^*(V_i)$, which correspond to δ_u and δ_u^*, cf. Definition 5. Then, given a trapezoid representation R of G, we may assume in Lemma 4 w.l.o.g. that $R(V_i) \ll_R T_u$, by possibly performing a vertical axis flipping of R. Thus, since $N_X(u) = N_X(u, V_i, V_j)$ for every $X \in \{0, 1, 2, 12\}$, Corollary 1 follows from Lemma 4.

Corollary 1. *Let G be a trapezoid graph, R be a representation of G, and u be a vertex of G, where $\delta_u^* \neq \emptyset$. Let V_i be the master component of u that corresponds to δ_u. If $R(V_i) \ll_R T_u$, then $N_X(u) = N_X(u, R)$ for every $X \in \{0, 1, 2, 12\}$.*

In the following, we state two auxiliary lemmas that will be used in the proof of Theorem 1.

Lemma 5. *Let G be a trapezoid graph and u be a vertex of G. Then, $N_2(u) \cup N_{12}(u) = \emptyset$ if and only if $\delta_u^* = \emptyset$.*

Proof. Suppose first that $\delta_u^* = \emptyset$. Then, clearly there exists no vertex $v \in N(u)$ adjacent to δ_u^*, and thus, $N_2(u) \cup N_{12}(u) = \emptyset$. Conversely, suppose that $N_2(u) \cup N_{12}(u) = \emptyset$, and assume that $\delta_u^* \neq \emptyset$. Let $\delta_u = V_i$ and $\delta_u^* = V_j$, where V_i is a master component of u and V_j is a maximal component of $D_u^*(V_i)$. If $N(V_j) = \emptyset$, then clearly $N(V_j) \subseteq N(V_i)$, and thus, $V_j \in D_u(V_i)$, which is a contradiction. Thus, $N(V_j) \neq \emptyset$, i.e. some vertices of $N(u)$ are adjacent to some vertices of V_j. Since $\delta_u^* = V_j$, it follows by Definition 6 that $N_2(u) \cup N_{12}(u) \neq \emptyset$, which is a contradiction. Thus, $\delta_u^* = \emptyset$. $\square$

Lemma 6. *Let G be a trapezoid graph and u be a vertex of G. If $\delta_u^* \neq \emptyset$, then $N_1(u) \cup N_{12}(u) \neq \emptyset$.*

Proof. Suppose that $\delta_u^* \neq \emptyset$. Let V_i be the master component that corresponds to δ_u, and V_j be the maximal component of $D_u^*(V_i)$ that corresponds to δ_u^*. Assume that $N_1(u) \cup N_{12}(u) = \emptyset$, i.e. no neighbor of u is adjacent to any vertex $v \in V_i$. It follows that $N(V_i) = \emptyset$. On the other hand, since $\delta_u^* \neq \emptyset$, we obtain by Lemma 5 that $N_2(u) \cup N_{12}(u) \neq \emptyset$. That is, some neighbors of u are adjacent to some vertices of V_j, i.e. $N(V_j) \neq \emptyset$. Therefore, $N(V_i) = \emptyset \subsetneqq N(V_j)$, and thus, $D_u(V_i) \subsetneqq D_u(V_j)$, i.e. $|D_u(V_i)| < |D_u(V_j)|$. This is a contradiction, since V_i is a master component of u. Thus, $N_1(u) \cup N_{12}(u) \neq \emptyset$. $\square$

2.3 A splitting algorithm

We define now the splitting of a vertex u of a trapezoid graph G, where $\delta_u^* \neq \emptyset$. Intuitively, if the graph G was given along with a specific trapezoid representation R, this would have meant that we replace the trapezoid T_u in R by its two lines $l(T_u)$ and $r(T_u)$.

Definition 7. *Let G be a trapezoid graph and u be a vertex of G, where $\delta_u^* \neq \emptyset$. The graph $G^{\#}(u)$ obtained by the* vertex splitting *of u is defined as follows:*

1. $V(G^{\#}(u)) = V(G) \setminus \{u\} \cup \{u_1, u_2\}$, *where u_1 and u_2 are the two new vertices,*
2. $E(G^{\#}(u)) = E[V(G) \setminus \{u\}] \cup \{u_1 x \mid x \in N_1(u)\} \cup \{u_2 x \mid x \in N_2(u)\} \cup \{u_1 x, u_2 x \mid x \in N_{12}(u)\}$.

The vertices u_1 and u_2 are the derivatives *of vertex u.*

Algorithm 1 Split-U

Input: A trapezoid graph G and a vertex subset $U = \{u_1, u_2, \ldots, u_k\}$, such that $\delta^*_{u_i} \neq \emptyset$ for all $i = 1, 2, \ldots, k$
Output: The permutation graph $G^{\#}(U)$

$\overline{U} \leftarrow V(G) \setminus U$; $H_0 \leftarrow G$
for $i = 1$ to k **do**
  $H_i \leftarrow H^{\#}_{i-1}(u_i)$ {H_i is obtained by the vertex splitting of u_i in H_{i-1}}
$G^{\#}(U) \leftarrow H_k[V(H_k) \setminus \overline{U}]$ {remove from H_k all unsplitted vertices}
return $G^{\#}(U)$

We state now the notion of a standard trapezoid representation with respect to a particular vertex, which will be used in the proof of Theorem 1.

Definition 8. *Let G be a trapezoid graph and u be a vertex of G, where $\delta^*_u \neq \emptyset$. A trapezoid representation R of G is* standard with respect to u*, if the following properties are satisfied:*

1. $l(T_u) \ll_R R(N_0(u) \cup N_2(u))$.
2. $R(N_0(u) \cup N_1(u)) \ll_R r(T_u)$.

Now, given a trapezoid graph G and a vertex subset $U = \{u_1, u_2, \ldots, u_k\}$, such that $\delta^*_{u_i} \neq \emptyset$ for every $i = 1, 2, \ldots, k$, Algorithm Split-U returns a graph $G^{\#}(U)$ by splitting every vertex of U exactly once. At every step, Algorithm Split-U splits a vertex of U, and finally, it removes all vertices of the set $V(G) \setminus U$, which have not been split.

Remark 2. As mentioned in Remark 1, a similar algorithm, called Split-All, was presented in [6]. We would like to emphasize here the following four differences between the two algorithms. First, that Split-All gets as input a sibling-free graph G (two vertices u, v of a graph G are called *siblings*, if $N[u] = N[v]$; G is called *sibling-free* if G has no pair of sibling vertices), while our Algorithm Split-U gets as an input any graph (though, we will use it only for trapezoid graphs), which may contain pairs of sibling vertices. Second, Split-All splits all the vertices of the input graph, while Split-U splits only a subset of them, which satisfy a special property. Third, the order of vertices that are split by Split-All depends on a certain property (inclusion-minimal neighbor set), while Split-U splits the vertices in an arbitrary order. Last, the main difference between these two algorithms is that they perform a different vertex splitting operation at every step, since Definitions 5 and 6 do not comply with the corresponding Definitions 4.1 and 4.2 of [6].

Theorem 1. *Let G be a trapezoid graph and $U = \{u_1, u_2, \ldots, u_k\}$ be a vertex subset of G, such that $\delta^*_{u_i} \neq \emptyset$ for every $i = 1, 2, \ldots, k$. Then, the graph $G^{\#}(U)$ obtained by Algorithm Split-U, is a permutation graph with $2k$ vertices. Furthermore, if G is acyclic, then $G^{\#}(U)$ is acyclic with respect to $\{u^1_i, u^2_i\}^k_{i=1}$, where u^1_i and u^2_i are the derivatives of u_i, $i = 1, 2, \ldots, k$.*

Proof. Let R be a trapezoid representation of G. In order to prove that the graph $G^{\#}(U)$ constructed by Algorithm Split-All is a permutation graph, we will construct from R a permutation representation $R^{\#}(U)$ of $G^{\#}(U)$. To this end, we will construct sequentially, for every $i = 1, 2, \ldots, k$, a standard trapezoid representation of H_{i-1} with respect to u_i, in which all derivatives u^1_j, u^2_j, $1 \leq j \leq i-1$, are represented by trivial trapezoids, i.e. lines.

Let $u = u_1$. If R is not a standard representation with respect to u, we construct first from R a trapezoid representation R' of G that satisfies the first condition of Definition 8. Then, we construct from R' a trapezoid representation R'' of G that satisfies also the second condition of Definition 8, i.e. R'' is a standard trapezoid representation R' of G with respect to u.

Let V_i be the master component of u that corresponds to δ_u. By possibly performing a vertical axis flipping of R, we may assume w.l.o.g. that $R(V_i) \ll_R T_u$. Furthermore, the sets $N_0(u)$, $N_1(u)$, $N_2(u)$, and $N_{12}(u)$ coincide by Lemma 1 with the sets $N_0(u,R)$, $N_1(u,R)$, $N_2(u,R)$, and $N_{12}(u,R)$, respectively. Recall that, by Definition 4, $D_1(u,R)$ and $D_2(u,R)$ denote the sets of trapezoids of R that lie completely to the left and to the right of T_u in R, respectively.

Let p_x and q_x the endpoints on L_1 and L_2, respectively, of the left line $l(T_x)$ of an arbitrary trapezoid T_x in R. Suppose that $N_0(u) \cup N_2(u) \neq \emptyset$. Let p_v and q_w be the leftmost endpoints on L_1 and L_2, respectively, of the trapezoids of $N_0(u) \cup N_2(u)$, and suppose that $p_v < p_u$ and $q_w < q_u$. Note that, possibly, $v = w$. Then, all vertices x, for which T_x has an endpoint between p_v and p_u on L_1 (resp. between q_w and q_u on L_2) are adjacent to u. Indeed, suppose otherwise that $T_x \cap T_u = \emptyset$, for such a vertex x. Then, since $T_v \cap T_u \neq \emptyset$ (resp. $T_w \cap T_u \neq \emptyset$), it follows that $T_x \cap T_v \neq \emptyset$ (resp. $T_x \cap T_w \neq \emptyset$). However, since $T_x \cap T_u = \emptyset$ and T_x has an endpoint to the left of T_u in R, it follows that $T_x \ll_R T_u$, i.e. $x \in D_1(u,R)$, and thus, by Definition 3, $v \in N_1(u) \cup N_{12}(u)$ (resp. $w \in N_1(u) \cup N_{12}(u)$), which is a contradiction.

Consider now a vertex $z \in N_1(u) \cup N_{12}(u)$ with $l(T_z) \ll_R l(T_u)$, where $p_v < p_z < p_u$. Then, $q_z < q_w$. Indeed, suppose otherwise that $q_w < q_z$ (recall that all endpoints are assumed to be distinct). Then, since $z \in N_1(u) \cup N_{12}(u)$, there exists a vertex $x \in D_1(u,R)$, i.e. with $T_x \ll_R T_u$, such that $T_z \cap T_x \neq \emptyset$. Since $v, w \in N_0(u) \cup N_2(u)$, it follows that $T_v \cap T_x = \emptyset$ and $T_w \cap T_x = \emptyset$, and thus, $T_x \ll_R T_v$ and $T_x \ll_R T_w$. Therefore, since $p_v < p_z$ and $q_w < q_z$, we obtain that $T_x \ll_R T_z$, and thus, $T_z \cap T_x = \emptyset$, which is a contradiction. It follows that $q_z < q_w$. Moreover, z is adjacent to all vertices x in G, whose trapezoid T_x has an endpoint on L_1 between p_v and p_z, including p_v. Indeed, otherwise, $T_x \ll_R T_z$, and thus, $T_x \ll_R T_u$, since $l(T_z) \ll_R l(T_u)$. This is however a contradiction, since $x \in N(u)$, as we have proved above. Similarly, if $q_w < q_z < q_u$, then $p_z < p_v$ and z is adjacent to all vertices x in G, whose trapezoid T_x has an endpoint on L_2 between q_w and q_z, including q_w.

We construct now from R a new trapezoid representation R' of G as follows. First, for all vertices $z \in N_1(u) \cup N_{12}(u)$ with $l(T_z) \ll_R l(T_u)$, for which $p_v < p_z < p_u$ (and thus $q_z < q_w$), we move the endpoint p_z of $l(T_z)$ directly before p_v on L_1. In the sequel, for all vertices $z' \in N_1(u) \cup N_{12}(u)$ with $l(T_{z'}) \ll_R l(T_u)$, for which $q_w < q_{z'} < q_u$ (and thus $p_z < p_v$), we move the endpoint $q_{z'}$ of $l(T_{z'})$ directly before q_w on L_2. During the movement of all these lines $l(T_z)$ (resp. $l(T_{z'})$), we keep the same relative positions of their endpoints p_z on L_1 (resp. $q_{z'}$ on L_2) as in R, and thus we introduce no new line intersection among the lines of the trapezoids of G. Since all these vertices z (resp. z') are adjacent to all vertices x of G, whose trapezoid T_x has an endpoint on L_1 (resp. L_2) between p_v and p_z, including p_v (resp. between q_w and q_z, including q_w), these movements do not remove any adjacency from, and do not add any new adjacency to G.

Finally, we move both endpoints p_u and q_u of $l(T_u)$ directly before p_v and q_w on L_1 and L_2, respectively. Since u is adjacent to all vertices x, for which T_x has an endpoint between p_v and p_u on L_1, or between q_w and q_u on L_2 in R, the resulting representation R' is a trapezoid representation of G, in which the first condition of Definition 8 is satisfied. Since we moved all lines $l(T_z)$ and $l(T_{z'})$ to the left of T_v and T_w, R' has no additional line intersections than R. Moreover, note that for any line intersection of two lines a and b in R', the relative position of the endpoints of a and b on L_1 and L_2 remains the same as in R. In the case where $p_v > p_u$ (resp. $q_w > q_u$) we replace in the above construction p_v by p_u (resp. q_w by q_u), while in the case where $N_0(u) \cup N_2(u) = \emptyset$, we define $R' = R$. An example of the construction of R' is given in Figure 3. In this example, $v \in N_0(u)$, $w \in N_2(u)$, $z_1, z' \in N_1(u)$ and $z_2 \in N_{12}(u)$.

If R' is not a standard trapezoid representation with respect to u, then we move $r(T_u)$ to the right (similarly to the above), obtaining thus a trapezoid representation R'' of G, in which the second condition of Definition 8 is satisfied. Since during the construction of R'' from R'

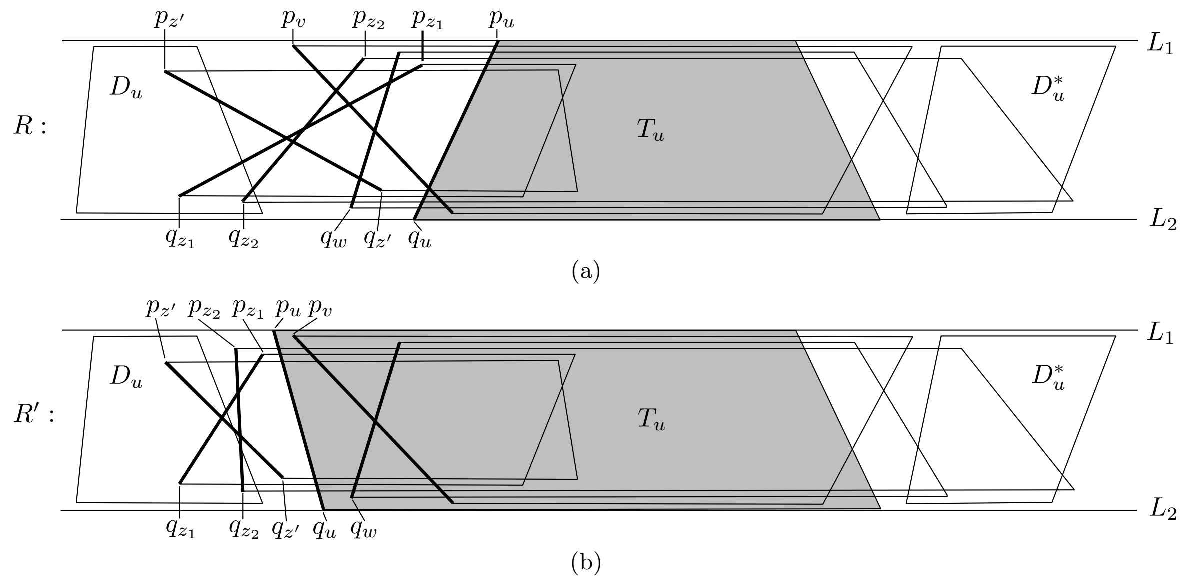

Fig. 3. The movement of the left line $l(T_u)$ of the trapezoid T_u, in order to construct a standard trapezoid representation with respect to u.

only the line $r(T_u)$, and other lines that lie completely to the right of $r(T_u)$, are moved to the right, the first condition of Definition 8 is satisfied for R'' as well. Thus, R'' is a standard representation of G with respect to u. Similarly to R', R'' has no additional line intersections than R. Moreover, for any line intersection of two lines a and b in R'', the relative position of the endpoints of a and b on L_1 and L_2 remains the same as in R.

Since R'' is standard with respect to u, the left line $l(T_u)$ of T_u in R'' intersects exactly with those trapezoids T_z, for which $z \in N_1(u) \cup N_{12}(u)$. On the other hand, the right line $r(T_u)$ of T_u in R'' intersects exactly with those trapezoids T_z, for which $z \in N_2(u) \cup N_{12}(u)$. Thus, if we replace in R'' the trapezoid T_u by the two trivial trapezoids (lines) $l(T_u)$ and $r(T_u)$, we obtain a trapezoid representation $R^{\#}(u)$ of the graph $G^{\#}(u)$ defined in Definition 7.

Consider now a vertex $v \in \{u_2, u_3, \ldots, u_k\}$. Due to the assumption, $\delta_v^* \neq \emptyset$ in G, before the vertex splitting of u, and thus, $N_2(v) \cup N_{12}(v) \neq \emptyset$ and $N_1(v) \cup N_{12}(v) \neq \emptyset$ in G by Lemmas 5 and 6. We will prove that $\delta_v^* \neq \emptyset$ in the trapezoid graph $G^{\#}(u)$ as well, after the vertex splitting of u. Due to Lemma 5, it suffices to show that $N_2(v) \cup N_{12}(v) \neq \emptyset$ in $G^{\#}(u)$. Let V_i be the master component of v in G that corresponds to δ_v, before the vertex splitting of u. We may assume w.l.o.g. that $R''(V_i) \ll_{R''} T_v$, by possibly performing a vertical axis flipping of R''. By Corollary 1, $N_1(v) \cup N_{12}(v) = N_1(v, R'') \cup N_{12}(v, R'')$ and $N_2(v) \cup N_{12}(v) = N_2(v, R'') \cup N_{12}(v, R'')$, i.e. these are the sets of neighbors of v in G, whose trapezoids intersect with the trapezoids of $D_1(v, R'')$ and $D_2(v, R'')$ in R'', respectively. Since $N_1(v, R'') \cup N_{12}(v, R'') \neq \emptyset$ and $N_2(v, R'') \cup N_{12}(v, R'') \neq \emptyset$ in G, and since $R^{\#}(u)$ is obtained from R'' by replacing the trapezoid T_u with the lines $l(T_u)$ and $r(T_u)$, it follows easily that $N_1(v, R^{\#}(u)) \cup N_{12}(v, R^{\#}(u)) \neq \emptyset$ and $N_2(v, R^{\#}(u)) \cup N_{12}(v, R^{\#}(u)) \neq \emptyset$ as well. Let V_k be the master component of v in $G^{\#}(u)$ that corresponds to δ_v, after the vertex splitting of u. If V_k lies to the left (resp. right) of T_v in $R^{\#}(u)$, then $N_2(v) \cup N_{12}(v)$ in $G^{\#}(u)$ equals to $N_2(v, R^{\#}(u)) \cup N_{12}(v, R^{\#}(u))$ (resp. to $N_1(v, R^{\#}(u)) \cup N_{12}(v, R^{\#}(u))$, by performing a vertical axis flipping of $R^{\#}(u)$). Therefore, $N_2(v) \cup N_{12}(v) \neq \emptyset$, and thus, $\delta_v^* \neq \emptyset$ in $G^{\#}(u)$, after the vertex splitting of u.

Applying iteratively the above construction for $u = u_i$, $i = 2, 3, \ldots, k$, i.e. by splitting sequentially all vertices of U exactly once, we obtain after k vertex splittings, and after removing from the resulting graph the vertices of $\overline{U} = V(G) \setminus U$, a trapezoid representation $R^{\#}(U)$ of the graph $G^{\#}(U)$ returned by Algorithm Split-U. Since every trapezoid T_u, $u \in U$, has been replaced by two trivial trapezoids, i.e. lines, in $R^{\#}(U)$, it follows that $G^{\#}(U)$ is a permutation graph with $2k$ vertices, and $R^{\#}(U)$ is a permutation representation of $G^{\#}(U)$.

Finally, suppose that R is an acyclic trapezoid representation of G. According to Definition 2, let P be the permutation graph with $2n$ vertices corresponding to the left and right lines of the trapezoids in R, R_P be the permutation representation of P induced by R, and $\{u_i^1, u_i^2\}$ be the vertices of P that correspond to the same vertex u_i of G, $i = 1, 2, \ldots, n$. Since R is an acyclic trapezoid representation of G, it follows by Definition 2 that R_P is an acyclic permutation representation with respect to $\{u_i^1, u_i^2\}_{i=1}^n$. That is, the simple directed graph F_{R_P} obtained (according to Definition 1) by merging u_i^1 and u_i^2 in P into a single vertex u_i, for every $i = 1, 2, \ldots, n$, has no directed cycle.

Since, during the construction of $R^{\#}(U)$, the trapezoid representation obtained after every vertex splitting has no additional line intersections than the previous one, it follows that $R^{\#}(U)$ has no additional line intersections than R. Moreover, for any line intersection of two lines a and b in $R^{\#}(U)$, the relative position of the endpoints of a and b on L_1 and L_2 remains the same as in R. Thus, the simple directed graph $F_{R^{\#}(U)}$ obtained (according to Definition 1) by merging u_i^1 and u_i^2 in $G^{\#}(U)$ into a single vertex u_i, for every $i = 1, 2, \ldots, k$, is a subdigraph of F_{R_P}. Therefore, since F_{R_P} has no directed cycle, $F_{R^{\#}(U)}$ has no directed cycle as well, i.e. $G^{\#}(U)$ is an acyclic permutation graph with respect to $\{u_i^1, u_i^2\}_{i=1}^k$. This completes the theorem. $\square$

3 The recognition of bounded tolerance graphs

In this section we provide a reduction from the *monotone-Not-All-Equal-3-SAT (monotone-NAE-3-SAT)* problem to the problem of recognizing whether a given graph is a bounded tolerance graph. A boolean formula ϕ is called *monotone* if no variable in ϕ is negated. Given a monotone boolean formula ϕ in conjunctive normal form with three literals in each clause (3-CNF), ϕ is *NAE-satisfiable* if there is a truth assignment of ϕ, such that every clause contains at least one true literal and at least one false one. The problem of deciding whether a given monotone 3-CNF formula ϕ is NAE-satisfiable is known to be NP-complete (see [27] for the NP-completeness of NAE-3-SAT[‡]). We can assume w.l.o.g. that each clause has three distinct literals, i.e. variables. Given a monotone 3-CNF formula ϕ, we construct in polynomial time a trapezoid graph H_ϕ, such that H_ϕ is a bounded tolerance graph if and only if ϕ is NAE-satisfiable. To this end, we construct first a permutation graph P_ϕ and a trapezoid graph G_ϕ.

3.1 The permutation graph P_ϕ

Consider a monotone 3-CNF formula $\phi = \alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_k$ with k clauses and n boolean variables $x_1, x_2, \ldots, x_n$, such that $\alpha_i = (x_{r_{i,1}} \vee x_{r_{i,2}} \vee x_{r_{i,3}})$ for $i = 1, 2, \ldots, k$, where $1 \leq r_{i,1} < r_{i,2} < r_{i,3} \leq n$. We construct the permutation graph P_ϕ, along with a permutation representation R_P of P_ϕ, as follows. Let L_1 and L_2 be two parallel lines and let $\theta(\ell)$ denote the angle of the line ℓ with L_2 in R_P. For every clause α_i, $i = 1, 2, \ldots, k$, we correspond to each of the literals, i.e. variables, $x_{r_{i,1}}$, $x_{r_{i,2}}$, and $x_{r_{i,3}}$ a pair of intersecting lines with endpoints on L_1 and L_2. Namely, we correspond to the variable $x_{r_{i,1}}$ the pair $\{a_i, c_i\}$, to $x_{r_{i,2}}$ the pair $\{e_i, b_i\}$ and to $x_{r_{i,3}}$ the pair $\{d_i, f_i\}$, respectively, such that $\theta(a_i) > \theta(c_i)$, $\theta(e_i) > \theta(b_i)$, $\theta(d_i) > \theta(f_i)$, and such that the lines a_i, c_i lie completely to the left of e_i, b_i in R_P, and e_i, b_i lie completely to the left of d_i, f_i in R_P, as it is illustrated in Figure 4. Denote the lines that correspond to

[‡]To reduce NAE-3-SAT to monotone-NAE-3-SAT, replace each variable x with two variables x_0 and x_1 (depending on whether x appears negated or not), add variables x_2, x_3, x_4, and add the clauses $(x_0 \vee x_1 \vee x_2)$, $(x_0 \vee x_1 \vee x_3)$, $(x_0 \vee x_1 \vee x_4)$, and $(x_2 \vee x_3 \vee x_4)$.

the variable $x_{r_{i,j}}$, $j = 1, 2, 3$, by $\ell^1_{i,j}$ and $\ell^2_{i,j}$, respectively, such that $\theta(\ell^1_{i,j}) > \theta(\ell^2_{i,j})$. That is, $(\ell^1_{i,1}, \ell^2_{i,1}) = (a_i, c_i)$, $(\ell^1_{i,2}, \ell^2_{i,2}) = (e_i, b_i)$, and $(\ell^1_{i,3}, \ell^2_{i,3}) = (d_i, f_i)$. Note that no line of a pair $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ intersects with a line of another pair $\{\ell^1_{i',j'}, \ell^2_{i',j'}\}$.

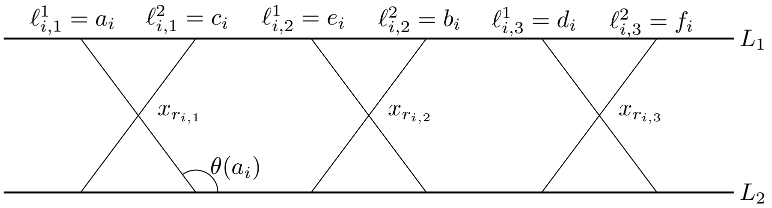

Fig. 4. The six lines of the permutation graph P_ϕ, which correspond to the clause $\alpha_i = (x_{r_{i,1}} \vee x_{r_{i,2}} \vee x_{r_{i,3}})$ of the boolean formula ϕ.

Denote by S_p, $p = 1, 2, \ldots, n$, the set of pairs $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ that correspond to the variable x_p, i.e. $r_{i,j} = p$. We order the pairs $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ such that any pair of S_{p_1} lies completely to the left of any pair of S_{p_2}, whenever $p_1 < p_2$, while the pairs that belong to the same set S_p are ordered arbitrarily. For two consecutive pairs $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ and $\{\ell^1_{i',j'}, \ell^2_{i',j'}\}$ in S_p, where $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ lies to the left of $\{\ell^1_{i',j'}, \ell^2_{i',j'}\}$, we add a pair $\{u^{i',j'}_{i,j}, v^{i',j'}_{i,j}\}$ of parallel lines that intersect both $\ell^1_{i,j}$ and $\ell^1_{i',j'}$, but no other line. Note that $\theta(\ell^1_{i,j}) > \theta(u^{i',j'}_{i,j})$ and $\theta(\ell^1_{i',j'}) > \theta(u^{i',j'}_{i,j})$, while $\theta(u^{i',j'}_{i,j}) = \theta(v^{i',j'}_{i,j})$. This completes the construction. Denote the resulting permutation graph by P_ϕ, and the corresponding permutation representation of P_ϕ by R_P. Observe that P_ϕ has n connected components, which are called *blocks*, one for each variable $x_1, x_2, \ldots, x_n$.

An example of the construction of P_ϕ and R_P from ϕ with $k = 3$ clauses and $n = 4$ variables is illustrated in Figure 5. In this figure, the lines $u^{i',j'}_{i,j}$ and $v^{i',j'}_{i,j}$ are drawn in bold.

The formula ϕ has $3k$ literals, and thus the permutation graph P_ϕ has $6k$ lines $\ell^1_{i,j}, \ell^2_{i,j}$ in R_P, one pair for each literal. Furthermore, two lines $u^{i',j'}_{i,j}, v^{i',j'}_{i,j}$ correspond to each pair of consecutive pairs $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ and $\{\ell^1_{i',j'}, \ell^2_{i',j'}\}$ in R_P, except for the case where these pairs of lines belong to different variables, i.e. when $r_{i,j} \neq r_{i',j'}$. Therefore, since ϕ has n variables, there are $2(3k - n) = 6k - 2n$ lines $u^{i',j'}_{i,j}, v^{i',j'}_{i,j}$ in R_P. Thus, R_P has in total $12k - 2n$ lines, i.e. P_ϕ has $12k - 2n$ vertices. In the example of Figure 5, $k = 3$, $n = 4$, and thus, P_ϕ has 28 vertices.

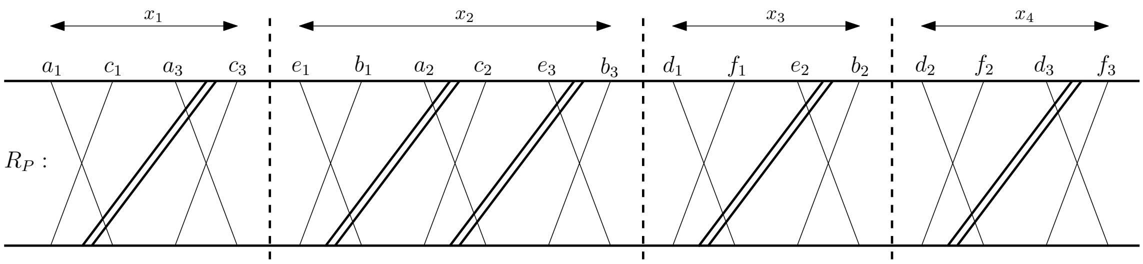

Fig. 5. The permutation representation R_P of the permutation graph P_ϕ for $\phi = \alpha_1 \wedge \alpha_2 \wedge \alpha_3 = (x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee x_4)$.

Let $m = 6k - n$, where $2m$ is the number of vertices in P_ϕ. We group the lines of R_P, i.e. the vertices of P_ϕ, into pairs $\{u^1_i, u^2_i\}^m_{i=1}$, as follows. For every clause α_i, $i = 1, 2, \ldots, k$, we group the lines $a_i, b_i, c_i, d_i, e_i, f_i$ into the three pairs $\{a_i, b_i\}$, $\{c_i, d_i\}$, and $\{e_i, f_i\}$. The remaining lines are grouped naturally according to the construction; namely, every two lines $\{u_{i,j}^{i',j'}, v_{i,j}^{i',j'}\}$ constitute a pair.

Lemma 7. *If the permutation graph P_ϕ is acyclic with respect to $\{u_i^1, u_i^2\}_{i=1}^m$ then the formula ϕ is NAE-satisfiable.*

Proof. Suppose that P_ϕ is acyclic with respect to $\{u_i^1, u_i^2\}_{i=1}^m$, and let R_0 be an acyclic permutation representation of P_ϕ with respect to $\{u_i^1, u_i^2\}_{i=1}^m$. Then, in particular, R_0 is acyclic with respect to $\{a_i, b_i\}, \{c_i, d_i\}, \{e_i, f_i\}$, for every $i = 1, 2, \ldots, k$. We will construct a truth assignment of the variables $x_1, x_2, \ldots, x_n$ that NAE-satisfies ϕ, as follows. For every $i = 1, 2, \ldots, k$, we define $x_{r_{i,1}} = 1$ if and only if $\theta(c_i) < \theta(a_i)$ in R_0, $x_{r_{i,2}} = 1$ if and only if $\theta(b_i) < \theta(e_i)$ in R_0, and $x_{r_{i,3}} = 1$ if and only if $\theta(f_i) < \theta(d_i)$ in R_0 (cf. Figure 6).

Note that this assignment is consistent; that is, all variables $x_{r_{i,j}}$ that correspond to the same x_k are assigned the same value. Indeed, the existence of the lines $u_{i,j}^{i',j'}, v_{i,j}^{i',j'}$ (cf. the bold lines in Figure 6(a)) forces all pairs of crossing lines $\{\ell_{i,j}^1, \ell_{i,j}^2\}$ in the same block to correspond to either 0 or 1 in the assignment.

Now, we show that in each clause α_i, $i = 1, 2, \ldots, k$, there exists at least one true and at least one false variable. For an arbitrary index $i = 1, 2, \ldots, k$, let P_i be the subgraph induced by the vertices $a_i, b_i, c_i, d_i, e_i, f_i$ in P_ϕ, and R_i be the permutation representation of P_i, which is induced by R_0. According to Definition 1, we construct the simple directed graph F_{R_i} by merging into a single vertex each of the pairs $\{a_i, b_i\}$, $\{c_i, d_i\}$ and $\{e_i, f_i\}$ of vertices of P_i. The arc directions of F_{R_i} are implied by the corresponding directions in Φ_{R_i} (or equivalently, in Φ_{R_0}). Then, since R_0 is acyclic with respect to $\{a_i, b_i\} \cup \{c_i, d_i\} \cup \{e_i, f_i\}$, so is R_i. Thus, it follows by Definition 1 that F_{R_i} has no directed cycle. Therefore, the edges $c_i a_i$, $b_i e_i$, and $f_i d_i$ of P_ϕ have such directions in Φ_{R_0} that it does not hold simultaneously $c_i a_i, b_i e_i, f_i d_i \in \Phi_{R_0}$, or $a_i c_i, e_i b_i, d_i f_i \in \Phi_{R_0}$. That is, it does not hold simultaneously $\theta(c_i) < \theta(a_i)$, $\theta(b_i) < \theta(e_i)$, and $\theta(f_i) < \theta(d_i)$, or $\theta(a_i) < \theta(c_i)$, $\theta(e_i) < \theta(b_i)$, and $\theta(d_i) < \theta(f_i)$ in R_0, respectively. Then, by the definition of the above truth assignment, it follows that it does not hold simultaneously $x_{r_{i,1}} = x_{r_{i,2}} = x_{r_{i,3}} = 1$, or $x_{r_{i,1}} = x_{r_{i,2}} = x_{r_{i,3}} = 0$, and therefore, the clause $\alpha_i = (x_{r_{i,1}} \vee x_{r_{i,2}} \vee x_{r_{i,3}})$ is NAE-satisfied. Finally, since this holds for every $i = 1, 2, \ldots, k$, ϕ is NAE-satisfiable. $\square$

For the formula ϕ of Figure 5, an example of an acyclic permutation representation R_0 of P_ϕ with respect to $\{u_i^1, u_i^2\}_{i=1}^m$, along with the corresponding transitive orientation Φ_{R_0} of P_ϕ, is illustrated in Figure 6. This transitive orientation corresponds to the NAE-satisfying truth assignment $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$ of ϕ. Similarly to Figure 5, the lines $u_{i,j}^{i',j'}$ and $v_{i,j}^{i',j'}$ are drawn in bold in Figure 6(a). Furthermore, for better visibility, the vertices that correspond to these lines are grouped in shadowed ellipses in Figure 6(b), while the arcs incident to them are drawn dashed.

3.2 The trapezoid graphs G_ϕ and H_ϕ

Let $\{u_i^1, u_i^2\}_{i=1}^m$ be the pairs of vertices in the constructed permutation graph P_ϕ and R_P be its permutation representation. We construct now from P_ϕ the trapezoid graph G_ϕ with m vertices $\{u_1, u_2, \ldots, u_m\}$, as follows. We replace in the permutation representation R_P for every $i = 1, 2, \ldots, m$ the lines u_i^1 and u_i^2 by the trapezoid T_{u_i}, which has u_i^1 and u_i^2 as its left and right lines, respectively. Let R_G be the resulting trapezoid representation of G_ϕ.

Finally, we construct from G_ϕ the trapezoid graph H_ϕ with $7m$ vertices, by adding to every trapezoid T_{u_i}, $i = 1, 2, \ldots, m$, six parallelograms $T_{u_{i,1}}, T_{u_{i,2}}, \ldots, T_{u_{i,6}}$ in the trapezoid representation R_G, as follows. Let ε be the smallest distance in R_G between two different endpoints on L_1, or on L_2. The right (resp. left) line of $T_{u_{1,1}}$ lies to the right (resp. left) of u_1^1, and it is parallel to it at distance $\frac{\varepsilon}{2}$. The right (resp. left) line of $T_{u_{1,2}}$ lies to the left of u_1^1, and it is parallel to it at distance $\frac{\varepsilon}{4}$ (resp. $\frac{3\varepsilon}{4}$). Moreover, the right (resp. left) line of $T_{u_{1,3}}$ lies to the left of u_1^1, and it is parallel to it at distance $\frac{3\varepsilon}{8}$ (resp. $\frac{7\varepsilon}{8}$). Similarly, the left (resp. right) line of $T_{u_{1,4}}$ lies to the left (resp. right) of u_1^2, and it is parallel to it at distance $\frac{\varepsilon}{2}$. The left (resp. right) line of $T_{u_{1,5}}$ lies to the right of u_1^2, and it is parallel to it at distance $\frac{\varepsilon}{4}$ (resp. $\frac{3\varepsilon}{4}$). Finally, the right (resp. left) line of $T_{u_{1,6}}$ lies to the right of u_1^2, and it is parallel to it at distance $\frac{3\varepsilon}{8}$ (resp. $\frac{7\varepsilon}{8}$), as illustrated in Figure 7.

After adding the parallelograms $T_{u_{1,1}}, T_{u_{1,2}}, \ldots, T_{u_{1,6}}$ to a trapezoid T_{u_1}, we update the smallest distance ε between two different endpoints on L_1, or on L_2 in the resulting representation, and we continue the construction iteratively for all $i = 2, \ldots, m$. Denote by H_ϕ the resulting trapezoid graph with $7m$ vertices, and by R_H the corresponding trapezoid representation. Note that in R_H, between the endpoints of the parallelograms $T_{u_{i,1}}$, $T_{u_{i,2}}$, and $T_{u_{i,3}}$ (resp. $T_{u_{i,4}}$, $T_{u_{i,5}}$, and $T_{u_{i,6}}$) on L_1 and L_2, there are no other endpoints of H_ϕ, except those of u_i^1 (resp. u_i^2), for every $i = 1, 2, \ldots, m$. Furthermore, note that R_H is standard with respect to u_i, for every $i = 1, 2, \ldots, m$. The following auxiliary lemma will be used in the proof of Theorem 2.

Lemma 8. *In the trapezoid graph H_ϕ, $\delta^*_{u_i} \neq \emptyset$ for every $i = 1, 2, \ldots, m$.*

Proof. Let $i \in \{1, 2, \ldots, m\}$. Recall that, by Definition 4, $D_1(u_i, R_H)$ (resp. $D_2(u_i, R_H)$) denotes the set of trapezoids of H_ϕ that lie completely to the left (resp. to the right) of T_{u_i} in R_H. In particular, $T_{u_{i,2}}, T_{u_{i,3}} \in D_1(u_i, R_H)$ and $T_{u_{i,5}}, T_{u_{i,6}} \in D_2(u_i, R_H)$. By the construction of R_H, it is easy to see that $T_{u_{i,2}} \cup T_{u_{i,3}}$ (resp. $T_{u_{i,5}} \cup T_{u_{i,6}}$) is the rightmost (resp. leftmost) connected component of $D_1(u_i, R_H)$ (resp. $D_2(u_i, R_H)$). Thus, $N(V_k) \subseteq N(\{u_{i,2}, u_{i,3}\})$ (resp. $N(V_\ell) \subseteq N(\{u_{i,5}, u_{i,6}\})$), for every connected component V_k (resp. V_ℓ) of $D_1(u_i, R_H)$ (resp. $D_2(u_i, R_H)$). Let V_p be the master component of u_i, such that $D_{u_i} = V_p$. Then, either $V_p = \{u_{i,2}, u_{i,3}\}$, or

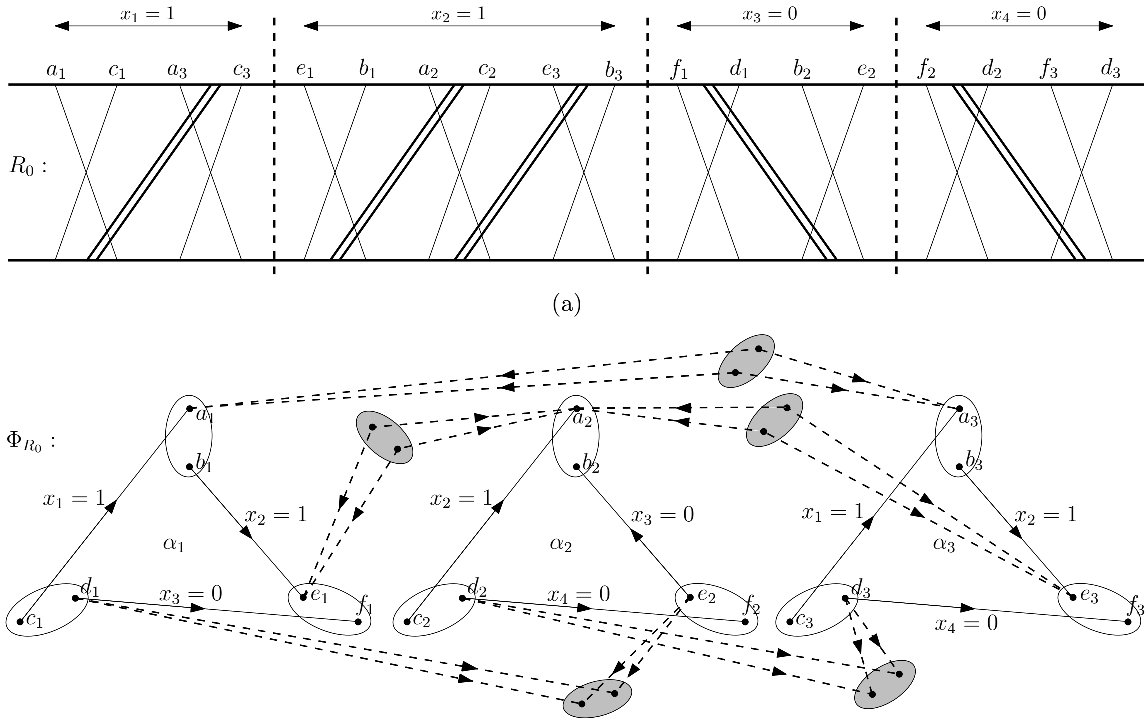

Fig. 6. The NAE-satisfying truth assignment $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$ of the formula ϕ of Figure 5: (a) an acyclic permutation representation R_0 of P_ϕ and (b) the corresponding transitive orientation Φ_{R_0} of P_ϕ.

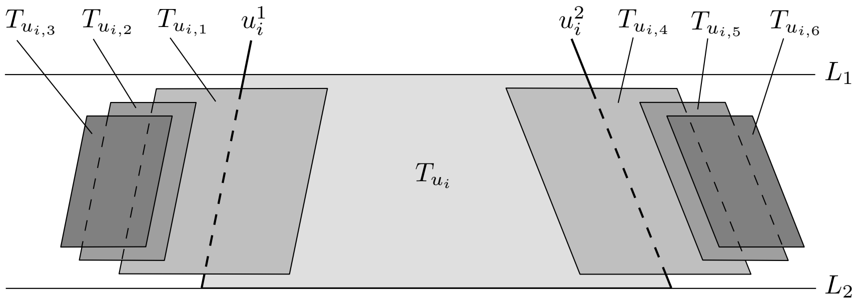

Fig. 7. The addition of the six parallelograms $T_{u_{i,1}}, T_{u_{i,2}}, \ldots, T_{u_{i,6}}$ to the trapezoid T_{u_i}, $i = 1, 2, \ldots, m$, in the construction of the trapezoid graph H_ϕ from G_ϕ.

$V_p = \{u_{i,5}, u_{i,6}\}$. In the case where $V_p = \{u_{i,2}, u_{i,3}\}$, we have $u_{i,4} \in N(\{u_{i,5}, u_{i,6}\}) \nsubseteq N(V_p)$, and thus $\{u_{i,5}, u_{i,6}\} \in \delta^*_{u_i}$. In the case where $V_p = \{u_{i,5}, u_{i,6}\}$, we have $u_{i,1} \in N(\{u_{i,2}, u_{i,3}\}) \nsubseteq N(V_p)$, and thus, $\{u_{i,2}, u_{i,3}\} \in \delta^*_{u_i}$. This proves the lemma. □

Theorem 2. *The formula ϕ is NAE-satisfiable if and only if the trapezoid graph H_ϕ is a bounded tolerance graph.*

Proof. Since a graph is a bounded tolerance graph if and only if it is a parallelogram graph [2], it suffices to prove that ϕ is NAE-satisfiable if and only if the trapezoid graph H_ϕ is a parallelogram graph.

($\Leftarrow$) Suppose that H_ϕ is a parallelogram graph, and let $U = \{u_1, u_2, \ldots, u_m\}$. Then, H_ϕ is an acyclic trapezoid graph by Lemma 1. Consider the permutation graph $H_\phi^\#(U)$ with $2m$ vertices, which is obtained by Algorithm Split-U on H_ϕ. Starting with the trapezoid representation R_H of H_ϕ, we obtain by the construction of Theorem 1 a permutation representation $R_H^\#(U)$ of $H_\phi^\#(U)$. Note that, since R_H is a standard trapezoid representation of H_ϕ with respect to every u_i, $i = 1, 2, \ldots, m$, the line u_i^1 (resp. u_i^2) of T_{u_i} is not moved during the construction of $R_H^\#(U)$ from R_H, for every $i = 1, 2, \ldots, m$. Therefore, $H_\phi^\#(U) = P_\phi$. On the other hand, since by Lemma 8 $\delta^*_{u_i} \neq \emptyset$ for every vertex $u_i \in U$, and since H_ϕ is an acyclic trapezoid graph, Theorem 1 implies that $H_\phi^\#(U) = P_\phi$ is an acyclic permutation graph with respect to $\{u_i^1, u_i^2\}_{i=1}^m$. Thus, ϕ is NAE-satisfiable by Lemma 7.

($\Rightarrow$) Conversely, suppose that ϕ has a NAE-satisfying truth assignment τ. We will construct first a permutation representation R_0 of P_ϕ, and then two trapezoid representations R_0' and R_0'' of G_ϕ and H_ϕ, respectively, as follows. Similarly to the representation R_P, the representation R_0 has n blocks, i.e. connected components, one for each variable $x_1, x_2, \ldots, x_n$. R_0 is obtained from R_P by performing a horizontal axis flipping of every block, which corresponds to a variable $x_p = 0$ in the truth assignment τ. Every other block, which corresponds to a variable $x_p = 1$ in the assignment τ, remains the same in R_0, as in R_P. Thus, $\theta(\ell^1_{i,j}) > \theta(\ell^2_{i,j})$ if $x_{r_{i,j}} = 1$ in τ, and $\theta(\ell^1_{i,j}) < \theta(\ell^2_{i,j})$ if $x_{r_{i,j}} = 0$ in τ, for every pair $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ of lines in R_0 (which correspond to the literal $x_{r_{i,j}}$ of the clause α_i in ϕ). An example of the construction of this representation R_0 of P_ϕ for the truth assignment $\tau = (1, 1, 0, 0)$ is illustrated in Figure 6(a).

Since τ is a NAE-satisfying truth assignment of ϕ, at least one literal is true and at least one is false in τ in every clause α_i, $i = 1, 2, \ldots, k$. Thus, there are six possible truth assignments for every clause, namely $(1, 1, 0)$, $(1, 0, 1)$, $(0, 1, 1)$, $(0, 0, 1)$, $(0, 1, 0)$, and $(1, 0, 0)$. For the first three ones, we can assign appropriate angles to the lines a_i, b_i, c_i, d_i, e_i, and f_i in the representation R_0, such that the relative positions of all endpoints in L_1 and L_2 remain unchanged, and such that a_i is parallel to b_i, c_i is parallel to d_i, and e_i is parallel to f_i, as illustrated in Figure 8. The last three truth assignments of α_i are the complement of the first three ones. Thus, by performing a horizontal axis flipping of the blocks in Figure 8, to which the lines a_i, b_i, c_i, d_i, e_i, and f_i belong, it is easy to see that for these assignments, we can also assign appropriate angles to these lines in the representation R_0, such that the relative positions of all endpoints in L_1 and L_2 remain unchanged, and such that a_i is parallel to b_i, c_i is parallel to d_i, and e_i is parallel to f_i.

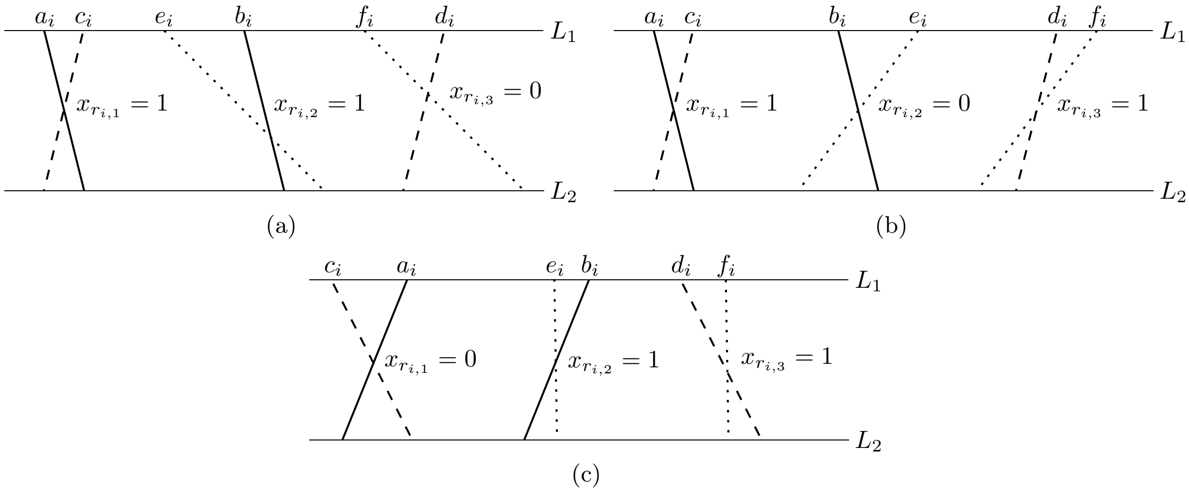

Fig. 8. The relative positions of the lines a_i, b_i, c_i, d_i, e_i, and f_i for the truth assignments (a) $(1,1,0)$, (b) $(1,0,1)$, and (c) $(0,1,1)$ of the clause α_i.

Recall that for every two consecutive pairs $\{\ell^1_{i,j}, \ell^2_{i,j}\}$ and $\{\ell^1_{i',j'}, \ell^2_{i',j'}\}$ of lines in R_P (resp. R_0), which belong to the same block, i.e. where $r_{i,j} = r_{i',j'}$, there are two parallel lines $u^{i',j'}_{i,j}, v^{i',j'}_{i,j}$ that intersect both $\ell^1_{i,j}$ and $\ell^1_{i',j'}$. Thus, after assigning the appropriate angles to the lines $\{\ell^1_{i,j}, \ell^2_{i,j}\}$, $i = 1,2,\ldots,k$, $j = 1,2,3$, we can clearly assign the appropriate angles to the lines $u^{i',j'}_{i,j}, v^{i',j'}_{i,j}$, such that the relative positions of all endpoints in L_1 and L_2 remain unchanged, and such that $u^{i',j'}_{i,j}$ remains parallel to $v^{i',j'}_{i,j}$. Summarizing, the lines u^1_i and u^2_i are parallel in R_0, for every $i = 1,2,\ldots,m$.

We construct now the trapezoid representation R'_0 of G_ϕ from the permutation representation R_0, by replacing for every $i = 1,2,\ldots,m$ the lines u^1_i and u^2_i by the trapezoid T_{u_i}, which has u^1_i and u^2_i as its left and right lines, respectively. Since R_0 is obtained by performing horizontal axis flipping of some blocks of R_P, and then changing the angles of the lines, while respecting the relative positions of the endpoints, R'_0 is indeed another trapezoid representation of G_ϕ than R_G. Since u^1_i is now parallel to u^2_i for every $i = 1,2,\ldots,m$, it follows clearly that R'_0 is a parallelogram representation, and thus, G_ϕ is a parallelogram graph.

Finally, we construct the trapezoid representation R''_0 of H_ϕ from R'_0, similarly to the construction of R_H from R_G. Namely, we add for every trapezoid T_{u_i}, $i = 1,2,\ldots,m$, six parallelograms $T_{u_{i,1}}, T_{u_{i,2}}, \ldots, T_{u_{i,6}}$, resulting in a trapezoid graph with $7m$ vertices. Since in R''_0 the parallelograms $T_{u_{i,1}}$, $T_{u_{i,2}}$, and $T_{u_{i,3}}$ (resp. $T_{u_{i,4}}$, $T_{u_{i,5}}$, and $T_{u_{i,6}}$) are sufficiently close to the left line u^1_i (resp. right line u^2_i) of T_{u_i}, $i = 1,2,\ldots,m$, and since between the endpoints of the parallelograms $T_{u_{i,1}}$, $T_{u_{i,2}}$, and $T_{u_{i,3}}$ (resp. $T_{u_{i,4}}$, $T_{u_{i,5}}$, and $T_{u_{i,6}}$) on L_1 and L_2, there are no other endpoints, it follows that R''_0 is indeed another trapezoid representation of H_ϕ than R_H. Finally, since R'_0 is a parallelogram representation, and since $T_{u_{i,1}}, T_{u_{i,2}}, \ldots, T_{u_{i,6}}$, $i = 1,2,\ldots,m$, are all parallelograms, R''_0 is also a parallelogram representation, and thus, H_ϕ is a parallelogram graph. □

Therefore, since monotone-NAE-3-SAT is NP-complete, the problem of recognizing bounded tolerance graphs is NP-hard. Moreover, since the recognition of bounded tolerance graphs lies in NP [16], we can summarize our results as follows.

Theorem 3. *Given a graph G, it is NP-complete to decide whether it is a bounded tolerance graph.*

4 The recognition of tolerance graphs

In this section we show that the reduction from the monotone-NAE-3-SAT problem to the problem of recognizing bounded tolerance graphs presented in Section 3, can be extended to the problem of recognizing general tolerance graphs. In particular, we prove that a given monotone 3-CNF formula ϕ is NAE-satisfiable if and only if the graph H_ϕ constructed in Section 3.2 is a tolerance graph.

We show that the reduction from the monotone-NAE-3-SAT problem to the problem of recognizing bounded tolerance graphs presented in Section 3, can be extended to the problem of recognizing general tolerance graphs. In particular, we prove that a given monotone 3-CNF formula ϕ is NAE-satisfiable if and only if the graph H_ϕ constructed in Section 3.2 is a tolerance graph.

4.1 Structural properties of tolerance graphs

In the following we assume w.l.o.g. that any tolerance graph has a tolerance representation, in which all tolerances are distinct and no two different intervals share an endpoint [12, 13]. We state now similarly to [13, 14] some definitions and lemmas concerning tolerance graphs. In a certain tolerance representation $\langle I, t \rangle$ of a tolerance graph $G = (V, E)$, a vertex v is called *bounded* if $t_v \leq |I_v|$; otherwise, v is called *unbounded*. An unbounded vertex v of G is called *inevitable* (for a certain tolerance representation), if v is not an isolated vertex, and if setting $t_v = |I_v|$ creates a new edge in the representation, that is, the representation is no longer a tolerance representation of G. A tolerance representation of G is called *inevitable unbounded*, if every unbounded vertex in this representation is inevitable. For an inevitable unbounded vertex v of G (for a certain tolerance representation), a vertex u is called a *hovering vertex* of v, if $uv \notin E$ and $I_v \subseteq I_u$. The next lemma follows easily from the above definitions.

Lemma 9. *There exists a hovering vertex u for every inevitable unbounded vertex v of the tolerance graph G (for a certain tolerance representation).*

Proof. Since v is an inevitable unbounded vertex, setting $t_v = |I_v|$ creates a new edge in G; let uv be such an edge. Then, clearly $I_u \cap I_v \neq \emptyset$. Since initially $uv \notin E$, it follows that $|I_u \cap I_v| < \min\{t_u, t_v\} \leq t_u$. Furthermore, since setting $t_v = |I_v|$ creates a new edge in G, we obtain that $\min\{t_u, |I_v|\} \leq |I_u \cap I_v| < t_u$, and thus, $|I_u \cap I_v| = |I_v|$, i.e. $I_v \subseteq I_u$. Therefore, since $uv \notin E$ and $I_v \subseteq I_u$, it follows that u is a hovering vertex of v. $\square$

Lemma 10 ([13]). *Every tolerance representation can be transformed into an inevitable one in $O(n^2)$ time.*

Lemma 11. *Let v be an inevitable unbounded vertex of a tolerance graph G and u be a hovering vertex of v, in a certain tolerance representation of G. Then, $N(v) \subseteq N(u)$ in G.*

Proof. Since v is an inevitable unbounded vertex, $N(v) = \emptyset$. Let $w \in N(v)$ be a neighbor of v in G. Since u is a hovering vertex of v, it follows that $uv \notin E$, and thus, $w \neq u$. Furthermore, since $vw \in E$, and since v is unbounded, we obtain that $\min\{t_v, t_w\} \leq |I_v \cap I_w| \leq |I_v| < t_v$, and thus, $t_w \leq |I_v \cap I_w|$. Then, since $I_v \subseteq I_u$, it follows that $|I_v \cap I_w| \leq |I_u \cap I_w|$, and thus, $t_w \leq |I_u \cap I_w|$, i.e. $w \in N(u)$. Therefore, $N(v) \subseteq N(u)$ in G. $\square$

4.2 The reduction

Consider now a monotone 3-CNF formula ϕ and the trapezoid graph H_ϕ constructed from ϕ in Section 3.2.

Lemma 12. *In the trapezoid graph H_ϕ, there are no two vertices u and v, such that $uv \notin E(H_\phi)$ and $N(v) \subseteq N(u)$ in H_ϕ.*

Proof. The proof is done by investigating all cases for a pair of non-adjacent vertices u, v. First, observe that, by the construction of H_ϕ from G_ϕ, we have $N[u_{i,2}] = N[u_{i,3}]$, $N[u_{i,1}] = N[u_{i,2}] \cup \{u_i\}$, $N[u_{i,5}] = N[u_{i,6}]$, and $N[u_{i,4}] = N[u_{i,5}] \cup \{u_i\}$.

Consider first two vertices u_i and u_k in H_ϕ, for some $i, k = 1, 2, \ldots, m$ and $i \neq k$. Then, by the construction of H_ϕ from G_ϕ, and since u_i and u_k are non-adjacent, $u_{i,1} \in N(u_i) \setminus N(u_k)$ and $u_{k,1} \in N(u_k) \setminus N(u_i)$. Consider next the vertices u_i and $u_{k,j}$, for some $i, k = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, 6$. If $i = k$, then $j \in \{2, 3, 5, 6\}$, since $u_{i,1}, u_{i,4} \in N(u_i)$. In the case where $j \in \{2, 3\}$, we have $u_{i,4} \in N(u_i) \setminus N(u_{k,j})$ and $u_{k,5-j} \in N(u_{k,j}) \setminus N(u_i)$, while in the case where $j \in \{5, 6\}$, we have $u_{i,1} \in N(u_i) \setminus N(u_{k,j})$ and $u_{k,11-j} \in N(u_{k,j}) \setminus N(u_i)$. Suppose that $i \neq k$. Then, it follows by the construction of H_ϕ from G_ϕ that $u_{i,1} \in N(u_i) \setminus N(u_{k,j})$. Furthermore, if $j \in \{1, 2, 3\}$ (resp. $j \in \{4, 5, 6\}$), then $u_{k,j'} \in N(u_{k,j}) \setminus N(u_i)$ for any index $j' \in \{1, 2, 3\} \setminus \{j\}$ (resp. $j' \in \{4, 5, 6\} \setminus \{j\}$).

Consider finally the vertices $u_{i,\ell}$ and $u_{k,j}$, for some $i, k = 1, 2, \ldots, m$ and $\ell, j = 1, 2, \ldots, 6$. If $i = k$, then w.l.o.g. $\ell \in \{1, 2, 3\}$ and $j \in \{4, 5, 6\}$, since $u_{i,\ell}$ and $u_{k,j}$ are non-adjacent. In this case, $u_{i,\ell'} \in N(u_{i,\ell}) \setminus N(u_{k,j})$ and $u_{k,j'} \in N(u_{k,j}) \setminus N(u_{i,\ell})$, for all indices $\ell' \in \{1, 2, 3\} \setminus \{\ell\}$ and $j' \in \{4, 5, 6\} \setminus \{j\}$. Suppose that $i \neq k$. If $j \in \{1, 2, 3\}$ (resp. $j \in \{4, 5, 6\}$), let j' be any index of $\{1, 2, 3\} \setminus \{j\}$ (resp. $\{4, 5, 6\} \setminus \{j\}$). Similarly, if $\ell \in \{1, 2, 3\}$ (resp. $\ell \in \{4, 5, 6\}$), let ℓ' be any index of $\{1, 2, 3\} \setminus \{\ell\}$ (resp. $\{4, 5, 6\} \setminus \{\ell\}$). Then, it follows by the construction of H_ϕ from G_ϕ that $u_{i,\ell'} \in N(u_{i,\ell}) \setminus N(u_{k,j})$ and $u_{k,j'} \in N(u_{k,j}) \setminus N(u_{i,\ell})$.

Therefore, for all possible choices of non-adjacent vertices u, v in the trapezoid graph H_ϕ, we have $N(u) \setminus N(v) \neq \emptyset$ and $N(v) \setminus N(u) \neq \emptyset$, which proves the lemma. □

Lemma 13. *If H_ϕ is a tolerance graph then it is a bounded tolerance graph.*

Proof. Suppose that H_ϕ is a tolerance graph, and consider a tolerance representation R of H_ϕ. Due to Lemma 10, we may assume w.l.o.g. that R is an inevitable unbounded representation. If R has no unbounded vertices, then we are done. Otherwise, there exists at least one inevitable unbounded vertex v in R, which has a hovering vertex u by Lemma 9, where $uv \notin E(H_\phi)$. Then, $N(v) \subseteq N(u)$ in H_ϕ by Lemma 11, which contradicts Lemma 12. Thus, there exists no unbounded vertex in R, i.e. H_ϕ is a bounded tolerance graph. □

Theorem 4. *The formula ϕ is NAE-satisfiable if and only if H_ϕ is a tolerance graph.*

Proof. Suppose that ϕ is NAE-satisfiable. Then, by Theorem 2, H_ϕ is a bounded tolerance graph, and thus, H_ϕ is a tolerance graph. Suppose conversely that H_ϕ is a tolerance graph. Then, by Lemma 13, H_ϕ is a bounded tolerance graph. Thus, ϕ is NAE-satisfiable by Theorem 2. □

Therefore, since monotone-NAE-3-SAT is NP-complete, the problem of recognizing tolerance graphs is NP-hard. Moreover, since the recognition of tolerance graphs lies in NP [16], and since H_ϕ is a trapezoid graph, we can summarize we can summarize our results in this section as follows.

Theorem 5. *Given a graph G, it is NP-complete to decide whether it is a tolerance graph. The problem remains NP-complete even if the given graph G is known to be a trapezoid graph.*

5 Concluding remarks

In this article we proved that both tolerance and bounded tolerance graph recognition problems are NP-complete, by providing a reduction from the monotone-NAE-3-SAT problem, thus answering a longstanding open question. Furthermore, our reduction implies that, given a trapezoid graph, it is NP-complete to decide whether this graph is a tolerance graph, or bounded tolerance, i.e. parallelogram graph. A *unit* interval representation is an interval representation in which all intervals have the same length. A *proper* interval representation is one in which no interval is properly contained in another. These terms can apply to both interval graphs and tolerance graphs. It is known that the subclasses of unit and proper interval graphs are equal [25], but the corresponding tolerance subclasses are different [2]. The recognition of unit and of proper tolerance graphs, as well as of any other subclass of tolerance graphs, except bounded tolerance and bounded bitolerance graphs (i.e. trapezoid graphs), remain interesting open problems [14].

References

1. S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.
2. K. P. Bogart, P. C. Fishburnb, G. Isaak, and L. Langley. Proper and unit tolerance graphs. *Discrete Applied Mathematics*, 60(1-3):99–117, 1995.
3. A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph classes: a survey.* Society for Industrial and Applied Mathematics, 1999.
4. A. H. Busch. A characterization of triangle-free tolerance graphs. *Discrete Applied Mathematics*, 154(3):471–477, 2006.
5. A. H. Busch and G. Isaak. Recognizing bipartite tolerance graphs in linear time. In *Proceedings of the 33rd International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 12–20, 2007.
6. F. Cheah and D. Corneil. On the structure of trapezoid graphs. *Discrete Applied Mathematics*, 66(2):109–133, 1996.
7. F. Cheah and D. Corneil, 2009. Personal communication.
8. S. Felsner. Tolerance graphs and orders. *Journal of Graph Theory*, 28:129–140, 1998.
9. P. Fishburn and W. Trotter. Split semiorders. *Discrete Mathematics*, 195:111–126, 1999.
10. M. Golumbic. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol. 57).* North-Holland Publishing Co., 2004.
11. M. Golumbic and C. Monma. A generalization of interval graphs with tolerances. In *Proceedings of the 13th Southeastern Conference on Combinatorics, Graph Theory and Computing, Congressus Numerantium 35*, pages 321–331, 1982.
12. M. Golumbic, C. Monma, and W. Trotter. Tolerance graphs. *Discrete Applied Mathematics*, 9(2):157–170, 1984.
13. M. Golumbic and A. Siani. Coloring algorithms for tolerance graphs: Reasoning and scheduling with interval constraints. In *Proceedings of the Joint International Conferences on Artificial Intelligence, Automated Reasoning, and Symbolic Computation (AISC/Calculemus)*, pages 196–207, 2002.
14. M. Golumbic and A. Trenk. *Tolerance Graphs.* Cambridge Studies in Advanced Mathematics, 2004.
15. M. Grötshcel, L. Lovász, and A. Schrijver. The Ellipsoid Method and its Consequences in Combinatorial Optimization. *Combinatorica*, 1:169–197, 1981.
16. R. B. Hayward and R. Shamir. A note on tolerance graph recognition. *Discrete Applied Mathematics*, 143(1-3):307–311, 2004.
17. G. Isaak, K. Nyman, and A. Trenk. A hierarchy of classes of bounded bitolerance orders. *Ars Combinatoria*, 69, 2003.
18. M. Kaufmann, J. Kratochvíl, K. A. Lehmann, and A. R. Subramanian. Max-tolerance graphs as intersection graphs: cliques, cycles, and recognition. In *Proceedings of the 17th annual ACM-SIAM symposium on Discrete Algorithms (SODA)*, pages 832–841, 2006.
19. J. M. Keil and P. Belleville. Dominating the complements of bounded tolerance graphs and the complements of trapezoid graphs. *Discrete Applied Mathematics*, 140(1-3):73–89, 2004.
20. L. Langley. *Interval tolerance orders and dimension.* PhD thesis, Dartmouth College, 1993.
21. L. Langley. A recognition algorithm for orders of interval dimension two. *Discrete Applied Mathematics*, 60(1-3):257–266, 1995.

22. T.-H. Ma and J. P. Spinrad. On the 2-chain subgraph cover and related problems. *Journal of Algorithms*, 17(2):251–268, 1994.
23. G. B. Mertzios, I. Sau, and S. Zaks. A New Intersection Model and Improved Algorithms for Tolerance Graphs. Technical report, RWTH Aachen University, March 2009.
24. G. Narasimhan and R. Manber. Stability and chromatic number of tolerance graphs. *Discrete Applied Mathematics*, 36:47–56, 1992.
25. F. Roberts. *Indifference graphs.* Proof Techniques in Graph Theory, Academic Press, New York, 139-146, 1969.
26. S. P. Ryan. Trapezoid order classification. *Order*, 15:341–354, 1998.
27. T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th annual ACM symposium on Theory of computing (STOC)*, pages 216–226, 1978.

Aachener Informatik-Berichte

This list contains all technical reports published during the past five years. A complete list of reports dating back to 1987 is available from http://aib.informatik.rwth-aachen.de/. To obtain copies consult the above URL or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

2004-01 * Fachgruppe Informatik: Jahresbericht 2003
2004-02 Benedikt Bollig, Martin Leucker: Message-Passing Automata are expressively equivalent to EMSO logic
2004-03 Delia Kesner, Femke van Raamsdonk, Joe Wells (eds.): HOR 2004 – 2nd International Workshop on Higher-Order Rewriting
2004-04 Slim Abdennadher, Christophe Ringeissen (eds.): RULE 04 – Fifth International Workshop on Rule-Based Programming
2004-05 Herbert Kuchen (ed.): WFLP 04 – 13th International Workshop on Functional and (Constraint) Logic Programming
2004-06 Sergio Antoy, Yoshihito Toyama (eds.): WRS 04 – 4th International Workshop on Reduction Strategies in Rewriting and Programming
2004-07 Michael Codish, Aart Middeldorp (eds.): WST 04 – 7th International Workshop on Termination
2004-08 Klaus Indermark, Thomas Noll: Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information
2004-09 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Parameterized Power Domination Complexity
2004-10 Zinaida Benenson, Felix C. Gärtner, Dogan Kesdogan: Secure Multi-Party Computation with Security Modules
2005-01 * Fachgruppe Informatik: Jahresbericht 2004
2005-02 Maximillian Dornseif, Felix C. Gärtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"
2005-03 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions
2005-04 Daniel Mölle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem
2005-05 Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots
2005-06 Simon Fischer, Berthold Vöcking: Adaptive Routing with Stale Information
2005-07 Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks
2005-08 Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut
2005-09 Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures
2005-10 Benedikt Bollig: Automata and Logics for Message Sequence Charts
2005-11 Simon Fischer, Berthold Vöcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture

2005-12 Neeraj Mittal, Felix Freiling, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed Systems

2005-13 Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments

2005-14 Felix C. Freiling, Sukumar Ghosh: Code Stabilization

2005-15 Uwe Naumann: The Complexity of Derivative Computation

2005-16 Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)

2005-17 Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)

2005-18 Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlußbericht des GI-Arbeitskreises "Features"

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht über den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Köln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Röglin, Berthold Vöcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-01 * Fachgruppe Informatik: Jahresbericht 2005

2006-02 Michael Weber: Parallel Algorithms for Verification of Large Systems

2006-03 Michael Maier, Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulation

2006-05 Uwe Naumann, Jean Utke, Patrick Heimbach, Chris Hill, Derya Ozyurt, Carl Wunsch, Mike Fagan, Nathan Tallent, Michelle Strout: Adjoint Code by Source Transformation with OpenAD/F

2006-06 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Divide-and-Color

2006-07 Thomas Colcombet, Christof Löding: Transforming structures by set interpretations

2006-08 Uwe Naumann, Yuxiao Hu: Optimal Vertex Elimination in Single-Expression-Use Graphs

2006-09 Tingting Han, Joost-Pieter Katoen: Counterexamples in Probabilistic Model Checking

2006-10 Mesut Günes, Alexander Zimmermann, Martin Wenig, Jan Ritzerfeld, Ulrich Meis: From Simulations to Testbeds - Architecture of the Hybrid MCG-Mesh Testbed

2006-11 Bastian Schlich, Michael Rohrbach, Michael Weber, Stefan Kowalewski: Model Checking Software for Microcontrollers

2006-12 Benedikt Bollig, Joost-Pieter Katoen, Carsten Kern, Martin Leucker: Replaying Play in and Play out: Synthesis of Design Models from Scenarios by Learning

2006-13 Wong Karianto, Christof Löding: Unranked Tree Automata with Sibling Equalities and Disequalities

2006-14 Danilo Beuche, Andreas Birk, Heinrich Dreier, Andreas Fleischmann, Heidi Galle, Gerald Heller, Dirk Janzen, Isabel John, Ramin Tavakoli Kolagari, Thomas von der Maßen, Andreas Wolfram: Report of the GI Work Group "Requirements Management Tools for Product Line Engineering"

2006-15 Sebastian Ullrich, Jakob T. Valvoda, Torsten Kuhlen: Utilizing optical sensors from mice for new input devices

2006-16 Rafael Ballagas, Jan Borchers: Selexels: a Conceptual Framework for Pointing Devices with Low Expressiveness

2006-17 Eric Lee, Henning Kiel, Jan Borchers: Scrolling Through Time: Improving Interfaces for Searching and Navigating Continuous Audio Timelines

2007-01 * Fachgruppe Informatik: Jahresbericht 2006

2007-02 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, and Harald Zankl: SAT Solving for Termination Analysis with Polynomial Interpretations

2007-03 Jürgen Giesl, René Thiemann, Stephan Swiderski, and Peter Schneider-Kamp: Proving Termination by Bounded Increase

2007-04 Jan Buchholz, Eric Lee, Jonathan Klein, and Jan Borchers: coJIVE: A System to Support Collaborative Jazz Improvisation

2007-05 Uwe Naumann: On Optimal DAG Reversal

2007-06 Joost-Pieter Katoen, Thomas Noll, and Stefan Rieger: Verifying Concurrent List-Manipulating Programs by LTL Model Checking

2007-07 Alexander Nyßen, Horst Lichter: MeDUSA - MethoD for UML2-based Design of Embedded Software Applications

2007-08 Falk Salewski and Stefan Kowalewski: Achieving Highly Reliable Embedded Software: An empirical evaluation of different approaches

2007-09 Tina Kraußer, Heiko Mantel, and Henning Sudbrock: A Probabilistic Justification of the Combining Calculus under the Uniform Scheduler Assumption

2007-10 Martin Neuhäußer, Joost-Pieter Katoen: Bisimulation and Logical Preservation for Continuous-Time Markov Decision Processes

2007-11 Klaus Wehrle (editor): 6. Fachgespräch Sensornetzwerke

2007-12 Uwe Naumann: An L-Attributed Grammar for Adjoint Code

2007-13 Uwe Naumann, Michael Maier, Jan Riehme, and Bruce Christianson: Second-Order Adjoints by Source Code Manipulation of Numerical Programs

2007-14 Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch: OpenAD/F: A Modular, Open-Source Tool for Automatic Differentiation of Fortran Codes

2007-15 Volker Stolz: Temporal assertions for sequential and concurrent programs

2007-16 Sadeq Ali Makram, Mesut Güneç, Martin Wenig, Alexander Zimmermann: Adaptive Channel Assignment to Support QoS and Load Balancing for Wireless Mesh Networks

2007-17 René Thiemann: The DP Framework for Proving Termination of Term Rewriting

2007-18 Uwe Naumann: Call Tree Reversal is NP-Complete

2007-19 Jan Riehme, Andrea Walther, Jörg Stiller, Uwe Naumann: Adjoints for Time-Dependent Optimal Control

2007-20 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf: Three-Valued Abstraction for Probabilistic Systems

2007-21 Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre: Compositional Modeling and Minimization of Time-Inhomogeneous Markov Chains

2007-22 Heiner Ackermann, Paul W. Goldberg, Vahab S. Mirrokni, Heiko Röglin, and Berthold Vöcking: Uncoordinated Two-Sided Markets

2008-01 * Fachgruppe Informatik: Jahresbericht 2007

2008-02 Henrik Bohnenkamp, Marielle Stoelinga: Quantitative Testing

2008-03 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, Harald Zankl: Maximal Termination

2008-04 Uwe Naumann, Jan Riehme: Sensitivity Analysis in Sisyphe with the AD-Enabled NAGWare Fortran Compiler

2008-05 Frank G. Radmacher: An Automata Theoretic Approach to the Theory of Rational Tree Relations

2008-06 Uwe Naumann, Laurent Hascoet, Chris Hill, Paul Hovland, Jan Riehme, Jean Utke: A Framework for Proving Correctness of Adjoint Message Passing Programs

2008-07 Alexander Nyßen, Horst Lichter: The MeDUSA Reference Manual, Second Edition

2008-08 George B. Mertzios, Stavros D. Nikolopoulos: The λ-cluster Problem on Parameterized Interval Graphs

2008-09 George B. Mertzios, Walter Unger: An optimal algorithm for the k-fixed-endpoint path cover on proper interval graphs

2008-10 George B. Mertzios, Walter Unger: Preemptive Scheduling of Equal-Length Jobs in Polynomial Time

2008-11 George B. Mertzios: Fast Convergence of Routing Games with Splittable Flows

2008-12 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, Verena Wolf: Abstraction for stochastic systems by Erlang's method of stages

2008-13 Beatriz Alarcón, Fabian Emmes, Carsten Fuhs, Jürgen Giesl, Raúl Gutiérrez, Salvador Lucas, Peter Schneider-Kamp, René Thiemann: Improving Context-Sensitive Dependency Pairs

2008-14 Bastian Schlich: Model Checking of Software for Microcontrollers

2008-15 Joachim Kneis, Alexander Langer, Peter Rossmanith: A New Algorithm for Finding Trees with Many Leaves

2008-16 Hendrik vom Lehn, Elias Weingärtner and Klaus Wehrle: Comparing recent network simulators: A performance evaluation study

2008-17 Peter Schneider-Kamp: Static Termination Analysis for Prolog using Term Rewriting and SAT Solving

2008-18 Falk Salewski: Empirical Evaluations of Safety-Critical Embedded Systems

2009-03 Alexander Nyßen: Model-Based Construction of Embedded Real-Time Software - A Methodology for Small Devices

2009-05 George B. Mertzios, Ignasi Sau, Shmuel Zaks: A New Intersection Model and Improved Algorithms for Tolerance Graphs

2009-07 Joachim Kneis, Alexander Langer, Peter Rossmanith: Derandomizing Non-uniform Color-Coding I

2009-08 Joachim Kneis, Alexander Langer: Satellites and Mirrors for Solving Independent Set on Sparse Graphs

* These reports are only available as a printed version.

Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.