Algorithmica (2004) 39: 95–126
DOI: 10.1007/s00453-003-1075-9

Algorithmica

Algorithms for P_4-Comparability Graph Recognition and Acyclic P_4-Transitive Orientation

Stavros D. Nikolopoulos[1] and Leonidas Palios[1]

Abstract. We consider two problems pertaining to P_4-comparability graphs, namely, the problem of recognizing whether a simple undirected graph is a P_4-comparability graph and the problem of producing an acyclic P_4-transitive orientation of a P_4-comparability graph. These problems have been considered by Hoàng and Reed who described $O(n^4)$- and $O(n^5)$-time algorithms for their solution, respectively, where n is the number of vertices of the input graph. Faster algorithms have recently been presented by Raschle and Simon, and by Nikolopoulos and Palios; the time complexity of these algorithms for either problem is $O(n + m^2)$, where m is the number of edges of the graph.

In this paper we describe $O(nm)$-time and $O(n + m)$-space algorithms for the recognition and the acyclic P_4-transitive orientation problems on P_4-comparability graphs. The algorithms rely on properties of the P_4-components of a graph, which we establish, and on the efficient construction of the P_4-components by means of the BFS-trees of the *complement* of the graph rooted at each of its vertices, without however explicitly computing the complement. Both algorithms are simple and use simple data structures.

Key Words. Perfectly orderable graph, Comparability graph, P_4-comparability graph, P_4-component, Recognition, P_4-transitive orientation.

1. Introduction. We consider simple non-trivial undirected graphs. Let G be such a graph. An *orientation* of the graph G is an antisymmetric directed graph obtained from G by assigning a direction to each of its edges. An orientation F of G is called *transitive* if it satisfies the following condition: if $\overrightarrow{ab} \in F$ and $\overrightarrow{bc} \in F$, then $\overrightarrow{ac} \in F$ [9], where by $\overrightarrow{uv}$ or $\overleftarrow{vu}$ we denote an edge directed from u to v. The transitivity of F implies directly that F does not contain a directed triangle; moreover, F does not contain any directed cycle, as can be shown by induction on the length of the cycle. The transitivity of F also implies that if abc is a chordless path on three vertices in G, then F contains the directed edges $\overrightarrow{ab}$ and $\overleftarrow{bc}$, or $\overleftarrow{ab}$ and $\overrightarrow{bc}$. An orientation of a graph G is called P_4-*transitive* if the orientation of every chordless path on four vertices of G is transitive, i.e., for such a path $abcd$, the path's edges are oriented in one of the following two ways: $\overrightarrow{ab}$, $\overleftarrow{bc}$, and $\overrightarrow{cd}$ or $\overleftarrow{ab}$, $\overrightarrow{bc}$, and $\overleftarrow{cd}$. It must be noted that, unlike transitivity, the P_4-transitivity does not imply acyclicity. The term borrows from the fact that a chordless path on four vertices is denoted by P_4.

A graph which admits a transitive orientation is called a *comparability graph* [8]–[10]; Figure 1(a) depicts a comparability graph. A graph is a P_4-*comparability graph* if it admits an acyclic P_4-transitive orientation [13], [14]. In light of these definitions, every

[1] Department of Computer Science, University of Ioannina, 45110 Ioannina, Greece. {stavros,palios}@cs.uoi.gr.

Received February 15, 2001; revised March 21, 2002. Communicated by D. P. Dobkin.
Online publication March 1, 2004.

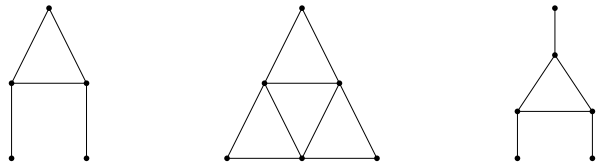

Fig. 1. (a) A comparability graph; (b) a P_4-comparability graph (which is not comparability); (c) a graph which is not P_4-comparability.

comparability graph is a P_4-comparability graph. In fact, the class of comparability graphs is a proper subset of the class of P_4-comparability graphs, as there exist P_4-comparability graphs which are not comparability; Figure 1(b) depicts such a graph, which is often referred to as the *pyramid*. The graph shown in Figure 1(c) is not a P_4-comparability graph. The class of P_4-comparability graphs was introduced by Hoáng and Reed, along with the classes of P_4-indifference, P_4-simplicial, and Raspail graphs, and all four classes were shown to be perfectly orderable [14].

The class of *perfectly orderable* graphs was introduced by Chvátal in the early 1980s [4]. These are the graphs for which there exists a *perfect order* on the set of their vertices. An order on the vertex set of a graph G is called *perfect* if for each subgraph H of G, the greedy algorithm (also known as the *first-fit* algorithm) computes an optimal coloring of H when processing the vertices of G in that order; see [4]. In the same paper, Chvátal showed that a graph is perfectly orderable if and only if there exists an acyclic orientation such that no P_4 $abcd$ of the graph has $\overrightarrow{ab}$ and $\overleftarrow{cd}$ (called *obstruction*). This directly implies that the P_4-comparability graphs are perfectly orderable, as they are defined to admit acyclic orientations that do not contain obstructions.

The class of perfectly orderable graphs is very important since a number of problems which are NP-complete in general can be solved in polynomial time on its members [2], [9], [12]; unfortunately, it is NP-complete to decide whether a graph admits a perfect order or, equivalently, an acyclic obstruction-free orientation [17]. Chvátal showed that the class of perfectly orderable graphs contains the comparability and the triangulated graphs [4]. It also contains a number of other classes of perfect graphs which are characterized by important algorithmic and structural properties; we mention the classes of 2-threshold, brittle, co-chordal, weak bipolarizable, distance hereditary, Meyniel $\cap$ co-Meyniel, P_4-sparse, etc. [3], [9]. Finally, since every perfectly orderable graph is strongly perfect [4], the class of perfectly orderable graphs is a subclass of the well-known class of perfect graphs.

Algorithms for many different problems on all the above-mentioned subclasses of perfectly orderable graphs are available in the literature. The comparability graphs in particular have been the focus of much research which culminated into efficient recognition and orientation algorithms [3], [9], [16]. On the other hand, the P_4-comparability graphs have not received as much attention, despite the fact that the definitions of the comparability and the P_4-comparability graphs rely on the same principles [7], [13], [14], [18], [19].

Our main objective is to study the recognition and acyclic P_4-transitive orientation problems on the class of P_4-comparability graphs. These problems have been addressed

by Hoàng and Reed who described polynomial time algorithms for their solution [13], [14]. The algorithms are based on detecting whether the input graph G contains a "homogeneous set" or a "good partition" and recursively solve the same problem on the graph that results from the input graph after contraction of one or two vertex sets into a single vertex each. The recognition and the orientation algorithms require $O(n^4)$ and $O(n^5)$ time, respectively, where n is the number of vertices of G. Recently, newer results on these problems were provided by Raschle and Simon [19]. Their algorithms work along the same lines, but they focus on the P_4-components of the graph. In particular, for a non-trivial P_4-component $\mathcal{C}$ of the input graph G, they compute the set R of vertices adjacent to some but not all the vertices of $\mathcal{C}$; depending on whether R is empty or not, they contract $\mathcal{C}$ into one or two (non-adjacent) vertices and they recursively solve the problem on the resulting graph. The time complexity of their algorithms for either problem is $O(n + m^2)$, where m is the number of edges of G, as it is dominated by the time to compute the P_4-components of G. Recently, Nikolopoulos and Palios described different $O(n + m^2)$-time algorithms for these problems [18]; their algorithms construct the P_4-components of the input graph G by means of the BFS-trees of G rooted at each of G's vertices.

In this paper we present $O(nm)$-time and $O(n + m)$-space algorithms for the recognition and the acyclic P_4-transitive orientation problems on P_4-comparability graphs. The stated time complexity may seem surprising in light of the fact that a graph on m edges may have as many as $\Theta(m^2)$ P_4s and that these algorithms, like previous ones, rely on the construction of the P_4-components of the input graph. Our approach avoids the computation of all the P_4s of the input graph G; instead, it computes all the P_3s that participate in P_4s of G. The computation takes advantage of the fact that the complement of a P_4 is also a P_4 and constructs these P_3s by means of the BFS-trees of the *complement* of G rooted at each of its vertices (without however explicitly computing the complement of G). The P_4-transitive orientation algorithm exploits structural relationships of the P_4-components of a graph, which we establish and which are interesting in their own right. Both algorithms are simple and use simple data structures, and can thus be easily used in practice.

The paper is structured as follows. In Section 2 we review the terminology that we use throughout the paper and we establish the theoretical framework on which our algorithms are based. We describe and analyze the recognition and orientation algorithms in Sections 3 and 4, respectively. We conclude with Section 5 which summarizes our results and addresses extensions and open problems.

2. Theoretical Framework. Let G be a simple non-trivial undirected graph; its vertex and edge sets are denoted by $V(G)$ and $E(G)$, respectively. A *path* in G is a sequence of vertices $v_0v_1 \cdots v_k$ such that $v_{i-1}v_i \in E(G)$ for $i = 1, 2, \ldots, k$; we say that this is a path from v_0 to v_k and that its *length* is k. A path is undirected or directed depending on whether G is an undirected or a directed graph. A path is called *simple* if none of its vertices occurs more than once; it is called *trivial* if its length is equal to 0. A path (simple path) $v_0v_1 \cdots v_k$ is called a *cycle* (*simple cycle*) of length $k+1$ if $v_0v_k \in E(G)$. A simple path (cycle) $v_0v_1 \cdots v_k$ is *chordless* if $v_iv_j \notin E(G)$ for any two non-consecutive vertices v_i, v_j in the path (cycle). Throughout the paper the chordless path (chordless

cycle, respectively) on n vertices is denoted by P_n (C_n, respectively). In particular, a chordless path on four vertices is denoted by P_4.

Let $abcd$ be a P_4. The vertices b and c are called the *midpoints* and the vertices a and d the *endpoints* of the P_4 $abcd$. The edge connecting the midpoints of a P_4 is called the *rib*; the other two edges (which are incident on the endpoints) are called the *wings*. For example, the edge bc is the rib and the edges ab and cd are the wings of the P_4 $abcd$. Two P_4s are called *adjacent* if they have an edge in common. The transitive closure of the adjacency relation is an equivalence relation on the set of P_4s of a graph G; the subgraphs of G spanned by the edges of the P_4s in the equivalence classes are the P_4-*components* of G. With slight abuse of terminology, we consider that an edge which does not belong to any P_4 belongs to a P_4-component by itself; such a component is called *trivial*. A P_4-component which is not trivial is called *non-trivial*; clearly a non-trivial P_4-component contains at least one P_4. For example, the graph of Figure 1(a) has one non-trivial P_4-component containing one P_4, and two trivial ones; the graph of Figure 1(b) has three non-trivial P_4-components containing one P_4 each; the graph of Figure 1(c) has one non-trivial P_4-component containing three P_4s which are pairwise adjacent. These and other simple examples may lead one to conjecture that the P_4-components of a P_4-comparability graph are comparability graphs. This is not true however, as indicated by the graph of Figure 2. The graph shown is a P_4-comparability graph and consists of two P_4-components: one contains the P_4s $a_1bc_1d_1$, $a_1bc_2d_1$, $a_1bc_2d_2$, $a_2bc_2d_1$, $a_2bc_2d_2$, and $a_2c_1c_2d_2$, and thus is the subgraph induced by $\{a_1, a_2, b, c_1, c_2, d_1, d_2\}$; the other contains the P_4s $rbpd_1$, $rbpd_2$, rc_1pa_1, rc_1pd_2, rc_2pa_1, and rc_2pa_2, and hence includes all the remaining edges. Note that the former component is not a comparability graph.

The definition of the P_4-components of a graph implies the following lemma.

LEMMA 2.1. *Let G be a graph and let $\mathcal{C}$ be a non-trivial P_4-component of G. Then*:

(i) *If ρ and ρ' are two P_4s which both belong to $\mathcal{C}$, then there exists a sequence ρ, ρ_1, $\ldots$, ρ_k, ρ' of adjacent P_4s in $\mathcal{C}$.*
(ii) *$\mathcal{C}$ is connected.*
(iii) *In any P_4-transitive orientation of $\mathcal{C}$, assigning an orientation on an edge of $\mathcal{C}$ forces the orientation on any other edge of $\mathcal{C}$.*

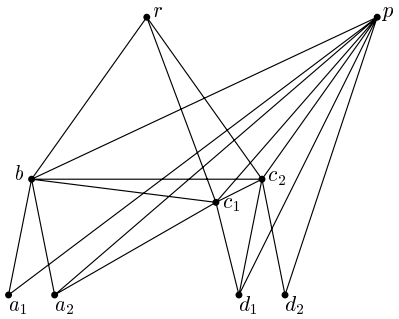

Fig. 2

Statement (iii) of Lemma 2.1 implies that if a P_4-component admits a P_4-transitive orientation F, then there exist precisely two possible orientations that may be assigned to it, namely, F and the orientation obtained from F after inversion of the direction of every edge in F.

The definition of a P_4-comparability graph requires that such a graph admits an acyclic P_4-transitive orientation. Interestingly, Hoàng and Reed [14] showed that in order to determine whether a graph is a P_4-comparability graph one can restrict one's attention to the P_4-components of the graph. In particular, what they proved [14, Theorem 3.1] can be paraphrased in terms of the P_4-components as follows:

LEMMA 2.2 [14]. *Let G be a graph such that each of its P_4-components admits an acyclic P_4-transitive orientation. Then G is a P_4-comparability graph.*

Although determining that each of the P_4-components of a graph admits an acyclic P_4-transitive orientation suffices to establish that the graph is P_4-comparability, the directed graph produced by placing the oriented P_4-components together may contain cycles. However, an acyclic P_4-transitive orientation of the entire graph can be obtained by inversion of the orientations of some of the P_4-components. Therefore, if one wishes to compute an acyclic P_4-transitive orientation of a P_4-comparability graph, one needs to detect directed cycles (if they exist) formed by edges belonging to more than one P_4-component and appropriately invert the orientation of one or more of these P_4-components. Fortunately, one does not need to consider arbitrarily long cycles as shown in the following lemma [14].

LEMMA 2.3 [14, Lemma 3.5]. *If a proper orientation of an interesting graph is cyclic, then it contains a directed triangle.*[2]

Given a non-trivial P_4-component $\mathcal{C}$ of a graph G, the set of vertices $V(G) - V(\mathcal{C})$ can be partitioned into three sets:

(i) R contains the vertices of $V(G) - V(\mathcal{C})$ which are adjacent to some (but not all) of the vertices in $V(\mathcal{C})$,
(ii) P contains the vertices of $V(G) - V(\mathcal{C})$ which are adjacent to all the vertices in $V(\mathcal{C})$, and
(iii) Q contains the vertices of $V(G) - V(\mathcal{C})$ which are not adjacent to any of the vertices in $V(\mathcal{C})$.

The adjacency relation is considered in terms of the input graph G.

In [19] Raschle and Simon showed that, given a non-trivial P_4-component $\mathcal{C}$ and a vertex $v \notin V(\mathcal{C})$, if v is adjacent to the midpoints of a P_4 of $\mathcal{C}$ and is not adjacent to its endpoints, then v does so with respect to every P_4 in $\mathcal{C}$ (that is, v is adjacent to the midpoints and not adjacent to the endpoints of every P_4 in $\mathcal{C}$). This implies that any vertex of G, which does not belong to $\mathcal{C}$ and is adjacent to at least one but not all the vertices in $V(\mathcal{C})$, is adjacent to the midpoints of all the P_4s in $\mathcal{C}$. Based on that, Raschle and Simon

[2] An orientation is proper if the orientation of every P_4 is transitive. A graph is interesting if the orientation of every P_4-component is acyclic.

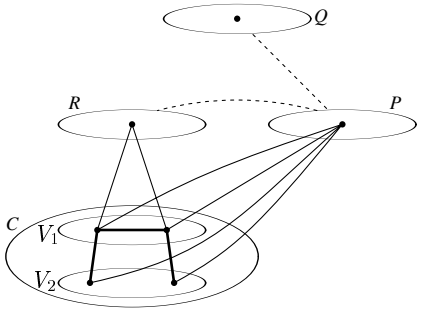

Fig. 3. Partition of the vertex set with respect to a separable P_4-component $\mathcal{C}$.

introduced the notion of a separable P_4-component—a non-trivial P_4-component $\mathcal{C}$ is *separable* if the set of midpoints and the set of endpoints of the P_4s of $\mathcal{C}$ define a partition of its vertex set $V(\mathcal{C})$—and showed that:

LEMMA 2.4 [19, Corollary 3.3]. *Let $\mathcal{C}$ be a non-trivial P_4-component and $R \neq \emptyset$. Then $\mathcal{C}$ is separable and every vertex in R is V_1-universal and V_2-null.*[3] *Moreover, no edge between R and Q exists.*

The set V_1 is the set of the midpoints of all the P_4s in $\mathcal{C}$, whereas the set V_2 is the set of endpoints. Figure 3 shows the partition of the vertices of a graph with respect to a separable P_4-component $\mathcal{C}$; the dashed segments between R and P and P and Q indicate that there may exist edges between pairs of vertices in the corresponding sets. Then a P_4 with at least one but not all its vertices in $V(\mathcal{C})$ must be a P_4 of one of the following types:

type (1)	vpq_1q_2	where	$v \in V(\mathcal{C}),\quad p \in P,\quad q_1, q_2 \in Q$;
type (2)	p_1vp_2q	where	$p_1 \in P,\quad v \in V(\mathcal{C}),\quad p_2 \in P,\quad q \in Q$;
type (3)	$p_1v_2p_2r$	where	$p_1 \in P,\quad v_2 \in V_2,\quad p_2 \in P,\quad r \in R$;
type (4)	$v_2pr_1r_2$	where	$v_2 \in V_2,\quad p \in P,\quad r_1, r_2 \in R$;
type (5)	rv_1pq	where	$r \in R,\quad v_1 \in V_1,\quad p \in P,\quad q \in Q$;
type (6)	rv_1pv_2	where	$r \in R,\quad v_1 \in V_1,\quad p \in P,\quad v_2 \in V_2$;
type (7)	$rv_1v_2v_2'$	where	$r \in R,\quad v_1 \in V_1,\quad v_2, v_2' \in V_2$;
type (8)	$v_1'rv_1v_2$	where	$r \in R,\quad v_1, v_1' \in V_1,\quad v_2 \in V_2$.

Raschle and Simon proved that neither a P_3 abc with $a \in V_1$ and $b,\ c \in V_2$ nor a $\overline{P_3}$ abc with $a,\ b \in V_1$ and $c \in V_2$ exists [19, Lemma 3.4], which implies that:

LEMMA 2.5. *Let $\mathcal{C}$ be a non-trivial P_4-component of a graph G. Then no P_4s of type* (7) *or* (8) *with respect to $\mathcal{C}$ exist.*

[3] For a set A of vertices, we say that a vertex v is A-universal if v is adjacent to every element of A; a vertex v is A-null if v is adjacent to no element of A.

Note that if the P_4-component $\mathcal{C}$ is not separable, then any P_4 with at least one but not all its vertices in $V(\mathcal{C})$ must be of type (1) or (2) only.

Raschle and Simon also proved the following interesting result regarding the P_4-components of a graph.

LEMMA 2.6 [19, Theorem 3.6]. *Two different P_4-components have different vertex sets.*

Additionally, we can show the following:

LEMMA 2.7. *Let $\mathcal{A}$ and $\mathcal{B}$ be two non-trivial P_4-components of a graph. If the component $\mathcal{A}$ contains an edge e both endpoints of which belong to the vertex set $V(\mathcal{B})$ of $\mathcal{B}$, then $V(\mathcal{A}) \subseteq V(\mathcal{B})$.*

PROOF. Suppose for contradiction that there exists a vertex $v \in V(\mathcal{A}) - V(\mathcal{B})$. We show that this implies that the P_4-component $\mathcal{A}$ contains a P_4 $\widehat{\rho}$ with a vertex not in $V(\mathcal{B})$ and an edge whose endpoints both belong to $V(\mathcal{B})$. Let ρ be the P_4 of $\mathcal{A}$ which contains the edge e. If ρ has a vertex which does not belong to $V(\mathcal{B})$, then $\widehat{\rho} = \rho$. Suppose now that all the vertices of ρ belong to $V(\mathcal{B})$. Since $v \in V(\mathcal{A})$, there exists a P_4 ρ' of $\mathcal{A}$ with v as a vertex. As both ρ and ρ' belong to $\mathcal{A}$, there is a sequence $\rho = \rho_0, \rho_1, \ldots, \rho_k = \rho'$ of adjacent P_4's in $\mathcal{A}$ (Lemma 2.1, statement (i)). Let ρ_i be the minimum-index P_4 in that sequence that has a vertex not in $V(\mathcal{B})$; clearly, ρ_i is well defined (because ρ' contains $v \notin V(\mathcal{B})$), and $i > 0$. By definition, ρ_i has a vertex not in $V(\mathcal{B})$; moreover, it is adjacent to ρ_{i-1}, all of whose vertices belong to $V(\mathcal{B})$, and it thus contains an edge whose endpoints both belong to $V(\mathcal{B})$. Then $\widehat{\rho} = \rho_i$.

Therefore, no matter whether the P_4 ρ contains a vertex not in $V(\mathcal{B})$ or not, we concluded that the P_4-component $\mathcal{A}$ indeed contains a P_4 $\widehat{\rho}$ as described. This however leads to a contradiction, since the P_4 $\widehat{\rho}$ would be of type (1)–(6) with respect to $\mathcal{B}$ (according to Lemma 2.5, no P_4s of type (7) or (8) exist), and yet no such P_4 has an edge whose endpoints both belong to $V(\mathcal{B})$. □

We consider a non-trivial P_4-component $\mathcal{C}$ of a graph G such that $V(\mathcal{C}) \subset V(G)$, and we let $S_{\mathcal{C}}$ be the set of non-trivial P_4-components of G which have a vertex in $V(\mathcal{C})$ and a vertex in $V(G) - V(\mathcal{C})$. Then, in light of Lemma 2.5, each component in $S_{\mathcal{C}}$ contains a P_4 of type (1)–(6), and thus we can partition the elements of $S_{\mathcal{C}}$ into two sets as follows:

- *P_4-components of type A*: the P_4 components, each of which contains at least one P_4 of type (1)–(5) with respect to $\mathcal{C}$;
- *P_4-components of type B*: the P_4-components which contain only P_4s of type (6) with respect to $\mathcal{C}$.

The following lemma establishes properties of P_4-components of type A.

LEMMA 2.8. *Let $\mathcal{C}$ be a non-trivial P_4-component of a graph G and suppose that the vertices in $V(G) - V(\mathcal{C})$ have been partitioned into sets R, P, and Q as described earlier in this section. If there exists an edge xv, where $x \in R \cup P$ and $v \in V(\mathcal{C})$, belonging to a P_4-component $\mathcal{A}$ of type A, then all the edges, which connect vertex x to a vertex in $V(\mathcal{C})$ and participate in a P_4 of G, belong to $\mathcal{A}$. Moreover, in a P_4-transitive orientation of $\mathcal{A}$ (assuming that $\mathcal{A}$ admits such an orientation), all these edges have the same orientation, i.e., they are all oriented either towards x or away from x.*

PROOF. Let xu be an edge of G connecting vertex x to a vertex u in $V(\mathcal{C})$. We show below that xu belongs to $\mathcal{A}$ and has the same orientation as xv.

If $\mathcal{C}$ is not separable, then $R = \emptyset$ (Lemma 2.4). Then the edge xv participates in P_4s of type (1) or (2) only. If it participates in a P_4 of type (1), say, in vxq_1q_2 ($q_1, q_2 \in Q$), then the path uxq_1q_2 is also a P_4 and therefore the edge xu belongs to $\mathcal{A}$ as well and has the same orientation as xv. Similarly, if xv participates in a P_4 of type (2), say, in p_1vp_2q (x is either p_1 or p_2, where $p_1, p_2 \in P$ and $q \in Q$), then the path p_1up_2q is also a P_4; again, the edge xu belongs to $\mathcal{A}$ and has the same orientation as xv.

We now consider the case that the P_4-component $\mathcal{C}$ is separable; let V_1 and V_2 be the sets of midpoints and of endpoints of the P_4s in $\mathcal{C}$. Since $u \in V(\mathcal{C})$, there exists a vertex w in $V(\mathcal{C})$ such that u and w are not adjacent and they do not both belong to V_1 or V_2; w is the vertex at distance 2 from v at a P_4 of $\mathcal{C}$ with v as a vertex. We distinguish the following cases:

Case (*a*): $x \in R$. Then $u \in V_1$, $w \in V_2$, and the edge xv participates in P_4s of type (5) or (6). If it participates in a P_4 of type (5), say, in $xvpq$ ($p \in P$, $q \in Q$), then the path $xupq$ is also a P_4 and therefore the edge xu belongs to $\mathcal{A}$ as well and has the same orientation as xv. Suppose now that xv participates in a P_4 of type (6), say, in $xvpv'$, where $p \in P$ and $v' \in V_2$ (Figure 4). Then, since xv belongs to the P_4-component $\mathcal{A}$, which is of type A and thus contains a P_4 of type (1)–(5), there exists a sequence S of adjacent P_4s from the P_4 $xvpv'$ to a P_4 of type (1)–(5) (Lemma 2.1, statement (i)). Without loss of generality, we may assume that all the P_4s in the sequence S except for the last one are P_4s of type (6); otherwise, we consider the prefix of the sequence up to the first P_4 of type (1)–(5). Let the sequence S be

$$xvpv' = r_1v_1p_1v_1',\ r_2v_2p_2v_2',\ \ldots,\ r_kv_kp_kv_k',\ \rho,$$

where $r_i \in R$, $v_i \in V_1$, $p_i \in P$, $v_i' \in V_2$, and ρ is a P_4 of type (1)–(5) adjacent to $r_kv_kp_kv_k'$. Clearly, all these P_4s belong to the component $\mathcal{A}$. Because each P_4 $r_iv_ip_iv_i'$ has one vertex from each one of four disjoint sets, the P_4s $r_iv_ip_iv_i'$ and $r_{i+1}v_{i+1}p_{i+1}v_{i+1}'$, which are adjacent, share an edge which is either a rib or a wing to both of them. So, the adjacency of $r_iv_ip_iv_i'$ and $r_{i+1}v_{i+1}p_{i+1}v_{i+1}'$ implies that $r_i = r_{i+1}$ and $v_i = v_{i+1}$, or $v_i = v_{i+1}$ and $p_i = p_{i+1}$, or $p_i = p_{i+1}$ and $v_i' = v_{i+1}'$. We now consider the sequence S'

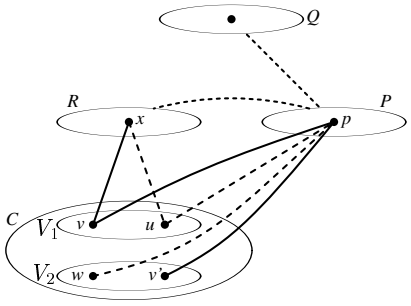

Fig. 4

of paths

$$xupw = r_1up_1w,\ r_2up_2w,\ \ldots,\ r_kup_kw.$$

It is not difficult to see that each of these paths is a P_4: $r_iu \in E(G)$, $p_iu \in E(G)$, $p_iw \in E(G)$, $uw \notin E(G)$, $r_iw \notin E(G)$, and, from the sequence S, $r_ip_i \notin E(G)$. Moreover, any two consecutive paths in S' are adjacent; note that the adjacency of $r_iv_ip_iv'_i$ and $r_{i+1}v_{i+1}p_{i+1}v'_{i+1}$ in S implies that $r_i = r_{i+1}$ or $p_i = p_{i+1}$ or both, which in turn implies that the P_4s r_iup_iw and $r_{i+1}up_{i+1}w$ are adjacent. Finally, because every element of P is $(V_1 \cup V_2)$-universal and every element of R is V_1-universal and V_2-null, the path ρ', which results from ρ after replacing v_k by u or v'_k by w (note that the general form of the P_4s of type (1)–(5) implies that ρ contains exactly one of v_k, v'_k), is a P_4 as well. Moreover, ρ' is adjacent to r_kup_kw (since ρ is adjacent to $r_kv_kp_kv'_k$), and ρ and ρ' are P_4s of the same type and have three vertices in common. Therefore, ρ and ρ' are adjacent, they belong to the same P_4-component $\mathcal{A}$ and they have corresponding orientations; then the edges r_kv_k and r_ku of their adjacent P_4s $r_kv_kp_kv'_k$ and r_kup_kw are oriented either both towards r_k or both away from it. In turn, the sequences S and S' of P_4s imply that the edges xv and xu belong to the same P_4-component and they are oriented either both towards x or both away from it, as desired.

Case (*b*): $x \in P$ *and* v, u *both belong to* V_1 *or both belong to* V_2. Then xv participates in P_4s of type (1)–(6). If it participates in a P_4, say, ρ, of type (1)-(5), then the path which results from ρ after replacing v by u is a P_4, is of the same type as ρ, and is adjacent to ρ. Therefore, the edges xv and xu belong to the same component $\mathcal{A}$ and have the same orientation. Suppose now that xv participates in a P_4 of type (6). We consider first the case where $v, u \in V_1$. Let v' be the vertex at distance 2 from v in any P_4 of $\mathcal{C}$ which has v as a vertex; clearly, $v' \in V_2$ and $vv' \notin E(G)$. Then the path $rvxv'$ is a P_4 and belongs to $\mathcal{A}$ (edge xv). Case (a) applies for the edges rv and ru, implying that they belong to $\mathcal{A}$ and they are oriented either both towards r or both away from it. Then so do the edges xv and xu because of the P_4s $rvxv'$ and $ruxw$. We work similarly in the second case, where $v, u \in V_2$; this time we consider the P_4s $rv'xv$ and $rwxu$.

Case (*c*): $x \in P$, $v \in V_1$, *and* $u \in V_2$. Then xv participates in a P_4, say, ρ, of type (1), (2), (5), or (6). If ρ is of type (1) or (2), then we work as in the first subcase of Case (b): replacing v by u in ρ yields a P_4, which together with ρ ensures that the edges xv and xu belong to the same P_4-component $\mathcal{A}$ and have the same orientation. If ρ is of type (5) or (6), i.e., of the form $rvxq$ or $rvxv'$, respectively ($r \in R$, $q \in Q$, $v' \in V_2$), then we consider the path $rwxu$ which is a P_4; note that $w \in V_1$ since $u \in V_2$. The lemma follows if we show that the edges rv and rw belong to the same P_4-component $\mathcal{A}$ and have the same orientation; this is established in Case (a) above.

Case (*d*): $x \in P$, $v \in V_2$, *and* $u \in V_1$. Since xu belongs to a P_4 then, according to Case (c), the edge xv belongs to the P_4-component to which xu belongs and has the same orientation as xu; thus, xv and xu belong to the P_4-component $\mathcal{A}$ and have the same orientation. □

The P_4-components of type B turn out to be really critical in the computation of a P_4-transitive orientation of a P_4-comparability graph. Indeed, as will be shown in

Section 4, if a directed triangle is formed when placing together the P_4-transitively oriented P_4-components of a P_4-comparability graph, then the triangle consists of three edges belonging to three P_4-components which are of type B with respect to one another. Lemma 2.9 suggests a test by means of which we can determine whether a P_4-component is of type B with respect to another P_4-component, Lemma 2.10 gives some general properties of a P_4-component of type B, and Lemmata 2.11 and 2.12 present cases where the "being of type B" relationship is symmetric and transitive, respectively; these lemmata will be useful in our orientation algorithm.

LEMMA 2.9. *Let $\mathcal{B}, \mathcal{C}$ be two non-trivial P_4-components of a graph such that $|V(\mathcal{B})| \geq |V(\mathcal{C})|$, and let $\beta = \sum_{v \in V(\mathcal{C})} d_{\mathcal{B}}(v)$, where $d_{\mathcal{B}}(v)$ denotes the number of edges of $\mathcal{B}$ which are incident upon v. Then $\mathcal{B}$ is of type B with respect to $\mathcal{C}$ if and only if $\beta = |E(\mathcal{B})|$.*

PROOF. Clearly, if $\mathcal{B}$ is of type B with respect to $\mathcal{C}$, then $\beta = |E(\mathcal{B})|$; note that each edge of a P_4 of type (6) with respect to $\mathcal{C}$ has exactly one of its endpoints in $V(\mathcal{C})$. Suppose now that $\beta = |E(\mathcal{B})|$; we will show that $\mathcal{B}$ is of type B with respect to $\mathcal{C}$. First, note that it is not possible that $V(\mathcal{B}) \cap V(\mathcal{C}) = \emptyset$; if this were the case, then $\beta = 0$ in contradiction to $\beta = |E(\mathcal{B})|$ in light of the fact that every non-trivial P_4-component contains at least one P_4 and thus contains at least three edges. Therefore, $V(\mathcal{B}) \cap V(\mathcal{C}) \neq \emptyset$. Additionally, it is impossible that $V(\mathcal{B}) \subseteq V(\mathcal{C})$; if so, then the inequality $|V(\mathcal{B})| \geq |V(\mathcal{C})|$ would imply that $V(\mathcal{B}) = V(\mathcal{C})$ and β would be equal to $2|E(\mathcal{B})|$ in contradiction to the fact that $\beta = |E(\mathcal{B})|$. Therefore, $V(\mathcal{B}) - V(\mathcal{C}) \neq \emptyset$. Since $V(\mathcal{B}) \cap V(\mathcal{C}) \neq \emptyset$ and $V(\mathcal{B}) - V(\mathcal{C}) \neq \emptyset$, $\mathcal{B}$ contains P_4s of type (1)–(6) (recall that Lemma 2.5 excludes P_4s of type (7) and (8)) and may contain P_4s none of whose vertices is a vertex in $V(\mathcal{C})$. The edges of the latter set of P_4s contribute nothing to the quantity β. On the other hand, the general form of the P_4s of type (1)–(6) indicates that the edges of such P_4s have at most one of their endpoints in $V(\mathcal{C})$, and thus contribute at most 1 to β each. Therefore, each edge of $\mathcal{B}$ contributes at most 1 to β. In order that $\beta = |E(\mathcal{B})|$, it is required that each edge contributes exactly 1. This is possible only if the edges participate in P_4s of type (6) with respect to $\mathcal{C}$; note that each P_4 of type (1)–(5) with respect to $\mathcal{C}$ contains at least one edge which is not incident upon any vertex of $\mathcal{C}$. Therefore, $\mathcal{B}$ is of type B with respect to $\mathcal{C}$. □

LEMMA 2.10. *Let $\mathcal{C}$ be a non-trivial P_4-component of a graph G, and let R, P, and Q be the partition sets of the vertices in $V(G) - V(\mathcal{C})$ as described earlier in this section. If $\mathcal{B}$ is a non-trivial P_4-component which is of type B with respect to $\mathcal{C}$, then*

(i) *both $\mathcal{B}$ and $\mathcal{C}$ are separable*;
(ii) *every edge of $\mathcal{B}$ has exactly one endpoint in $V(\mathcal{C})$*;
(iii) *for every P_4 $xyzw$ of $\mathcal{C}$, $x \in V(\mathcal{B})$ iff $z \in V(\mathcal{B})$*;
(iv) *at least one of the midpoints and at least one of the endpoints of every P_4 of $\mathcal{C}$ belongs to $V(\mathcal{B})$*;
(v) *for every vertex $r \in R \cap V(\mathcal{B})$ and for every vertex $u \in V_1(\mathcal{C}) \cap V(\mathcal{B})$, the edge ru belongs to $\mathcal{B}$; additionally, for every vertex $p \in P \cap V(\mathcal{B})$ and for every vertex $u \in V(\mathcal{C}) \cap V(\mathcal{B})$, the edge pu belongs to $\mathcal{B}$.*

PROOF. (i) Since $\mathcal{B}$ is of type B with respect to $\mathcal{C}$, then $R \neq \emptyset$; thus, $\mathcal{C}$ is separable in accordance with Lemma 2.4. Additionally, since all the P_4s of $\mathcal{B}$ are of type (6) with respect to $\mathcal{C}$, then the midpoints of all these P_4s are either midpoints of $\mathcal{C}$ or belong to P, whereas the endpoints are either endpoints of $\mathcal{C}$ or belong to R; thus, $\mathcal{B}$ is separable as well.

Since $\mathcal{C}$ is separable, in the rest of the proof we use $V_1(\mathcal{C})$ and $V_2(\mathcal{C})$ to denote the sets of midpoints and endpoints of the P_4s in $\mathcal{C}$, respectively.

(ii) Clearly true, because of the general form of the P_4s of type (6).

(iii) Clearly, $x \in V_2(\mathcal{C})$ and $z \in V_1(\mathcal{C})$. Suppose that $x \in V(\mathcal{B})$. Then there exists a P_4 of type (6) in $\mathcal{B}$ with x as a vertex; let it be $rx'px$, where $r \in R$ and $p \in P$. Then the path $rzpx$ is a P_4 and belongs to $\mathcal{B}$; thus, $z \in V(\mathcal{B})$. In a similar fashion, we can show that $z \in V(\mathcal{B})$ implies that $x \in V(\mathcal{B})$.

(iv) Let ρ be a P_4 of $\mathcal{B}$; obviously, ρ is of type (6) with respect to $\mathcal{C}$ and thus contains a vertex, say, v, which is a midpoint of a P_4 of $\mathcal{C}$. Then v belongs to a P_4 of $\mathcal{C}$; let it be $uvwz$. Clearly, the proposition holds for the P_4 $uvwz$: by definition $v \in V(\mathcal{B})$, which implies that $z \in V(\mathcal{B})$ in accordance with statement (iii). We will show that if it holds for the P_4 $abcd$ of $\mathcal{C}$, then it also holds for any P_4 $a'b'c'd'$ adjacent to $abcd$. Because $\mathcal{C}$ is separable (statement (i)), the two P_4s $abcd$ and $a'b'c'd'$ share an edge which is a rib or a wing to both of them; hence, without loss of generality, $a = a'$ and $b = b'$, or $b = b'$ and $c = c'$, or $c = c'$ and $d = d'$. Let b be the midpoint of the P_4 $abcd$ which belongs to $V(\mathcal{B})$; we will show that $b', d' \in V(\mathcal{B})$. We distinguish the following cases:

$a' = a$ and $b' = b$, or $b' = b$ and $c' = c$. Trivially, $b' \in V(\mathcal{B})$. Moreover, statement (iii) implies that $d' \in V(\mathcal{B})$.

$c' = c$ and $d' = d$. Trivially, $d' \in V(\mathcal{B})$; then, statement (iii) implies that $b' \in V(\mathcal{B})$.

Since for every P_4 ρ' of $\mathcal{C}$, there exists a sequence of adjacent P_4s from the P_4 $uvwz$ to ρ' (Lemma 2.1, statement (i)), the lemma follows.

(v) Because $r \in R \cap V(\mathcal{B})$ and the P_4-component $\mathcal{B}$ is of type B with respect to $\mathcal{C}$, $\mathcal{B}$ contains a P_4 $rvpv'$, where $v \in V_1(\mathcal{C})$, $p \in P$, and $v' \in V_2(\mathcal{C})$. Additionally, the fact that $u \in V_1(\mathcal{C}) \cap V(\mathcal{B})$ implies that $\mathcal{B}$ contains a P_4 $r'up'u'$, where $r' \in R$, $p' \in P$, and $u' \in V_2(\mathcal{C})$. Since both these P_4s belong to $\mathcal{B}$, there exists a sequence of adjacent P_4s (all of which are of type (6) with respect to $\mathcal{C}$) from $r'up'u'$ to $rvpv'$; let this sequence be

$$r'up'u' = r_1v_1p_1v_1',\ r_2v_2p_2v_2',\ \ldots,\ r_{k-1}v_{k-1}p_{k-1}v_{k-1}',\ r_kv_kp_kv_k' = rvpv'.$$

The adjacency of these P_4s implies that not both $r_i \neq r_{i+1}$ and $p_i \neq p_{i+1}$. Then the sequence

$$r'up'u',\ r_2up_2u',\ \ldots,\ r_{k-1}up_{k-1}u',\ rupu'$$

is a sequence of adjacent P_4s which belong to the P_4-component $\mathcal{B}$. Therefore, the edge ru belongs to $\mathcal{B}$. The proof for the edges pu, where $p \in P \cap V(\mathcal{B})$ and $u \in V(\mathcal{C}) \cap V(\mathcal{B})$, is similar. □

LEMMA 2.11. *Let $\mathcal{A}, \mathcal{B}$, and $\mathcal{C}$ be three distinct non-trivial P_4-components of a graph G such that $\mathcal{A}$ and $\mathcal{B}$ are of type B with respect to $\mathcal{C}$ and $(V(\mathcal{A}) \cap V(\mathcal{B})) - V(\mathcal{C}) \neq \emptyset$.*

Then

(i) $V(\mathcal{A}) - V(\mathcal{C}) = V(\mathcal{B}) - V(\mathcal{C})$;
(ii) *the sets* $V(\mathcal{A}) \cap V(\mathcal{C})$ *and* $V(\mathcal{B}) \cap V(\mathcal{C})$ *partition* $V(\mathcal{C})$; *in particular, the sets* $V_1(\mathcal{A}) \cap V_1(\mathcal{C})$ *and* $V_1(\mathcal{B}) \cap V_1(\mathcal{C})$ *partition* $V_1(\mathcal{C})$, *where by* $V_1(\mathcal{A})$, $V_1(\mathcal{B})$, *and* $V_1(\mathcal{C})$ *we denote the sets of midpoints of the* P_4*s in the* P_4*-components* $\mathcal{A}$, $\mathcal{B}$, *and* $\mathcal{C}$, *respectively*;
(iii) $\mathcal{C}$ *is of type B with respect to* $\mathcal{A}$ *and with respect to* $\mathcal{B}$;
(iv) $\mathcal{A}$ *is of type B with respect to* $\mathcal{B}$ *and vice versa*.

PROOF. Note that since the P_4-components $\mathcal{A}$ and $\mathcal{B}$ are of type B with respect to $\mathcal{C}$, Lemma 2.10 (statement (i)) implies that all three P_4-components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ are separable, and therefore the sets of their midpoints and endpoints are well defined. Below, for a separable P_4-component $\mathcal{K}$, the sets $V_1(\mathcal{K})$ and $V_2(\mathcal{K})$ denote the sets of midpoints and endpoints of the P_4s of $\mathcal{K}$, and the sets $R(\mathcal{K})$ and $P(\mathcal{K})$ the partition sets of the vertices of $V(G) - V(\mathcal{K})$ as described earlier.

(i) In order to prove that $V(\mathcal{A}) - V(\mathcal{C}) = V(\mathcal{B}) - V(\mathcal{C})$, it suffices to show that $V(\mathcal{B}) - V(\mathcal{C}) \subseteq V(\mathcal{A}) - V(\mathcal{C})$; then, by symmetry of the P_4-components $\mathcal{A}$ and $\mathcal{B}$ with respect to $\mathcal{C}$, $V(\mathcal{A}) - V(\mathcal{C}) \subseteq V(\mathcal{B}) - V(\mathcal{C})$ which implies the desired equality. Because $(V(\mathcal{A}) \cap V(\mathcal{B})) - V(\mathcal{C}) \neq \emptyset$, there exist vertices in $R(\mathcal{C}) \cup P(\mathcal{C})$ belonging to $V(\mathcal{A}) \cap V(\mathcal{B})$; let $r \in R(\mathcal{C})$ be such a vertex (the case for a vertex in $P(\mathcal{C})$ is similar). Then, since the P_4-components $\mathcal{A}$ and $\mathcal{B}$ are of type B with respect to $\mathcal{C}$, there exist P_4s $rupw$ and $ru'p'w'$ belonging to $\mathcal{A}$ and $\mathcal{B}$, respectively, where $u, u' \in V_1(\mathcal{C})$, $p, p' \in P(\mathcal{C})$, and $w, w' \in V_2(\mathcal{C})$; then the path $ru'pw'$ is a P_4 and belongs to $\mathcal{B}$ as well.

Let $x \in R(\mathcal{C}) \cap V(\mathcal{B})$ and let $xvyv'$ be a P_4 in $\mathcal{B}$ with x as a vertex. Then there exists a sequence of adjacent P_4s from $ru'pw'$ to $xvyv'$:

$$ru'pw' = r_1v_1p_1v_1',\ r_2v_2p_2v_2',\ \ldots,\ r_\ell v_\ell p_\ell v_\ell' = xvyv'.$$

Then the sequence

$$rupw = r_1up_1w,\ r_2up_2w,\ \ldots,\ r_{\ell-1}up_{\ell-1}w,\ xuyw$$

is a sequence of adjacent P_4s (probably with duplicates) belonging to $\mathcal{A}$. Thus, $x \in V(\mathcal{A})$. A similar approach establishes that if $x \in P(\mathcal{C}) \cap V(\mathcal{B})$, then $x \in V(\mathcal{A})$, which yields that $V(\mathcal{B}) - V(\mathcal{C}) \subseteq V(\mathcal{A}) - V(\mathcal{C})$.

(ii) We first show that the sets $V(\mathcal{A}) \cap V(\mathcal{C})$ and $V(\mathcal{B}) \cap V(\mathcal{C})$ are disjoint. Indeed, if there existed a vertex x in the intersection of these two sets, then the edge xy for any y in $V(\mathcal{A}) - V(\mathcal{C})$, which is not empty and is equal to $V(\mathcal{B}) - V(\mathcal{C})$ by statement (i), would belong to both $\mathcal{A}$ and $\mathcal{B}$ in accordance with Lemma 2.10 (statement (v)); however, this contradicts the fact that $\mathcal{A} \neq \mathcal{B}$. Proving that the union of $V(\mathcal{A}) \cap V(\mathcal{C})$ and $V(\mathcal{B}) \cap V(\mathcal{C})$ is equal to $V(\mathcal{C})$ follows from Lemma 2.10 (statement (iv)) and the fact that the two sets are disjoint.

The above property of the sets $V(\mathcal{A}) \cap V(\mathcal{C})$ and $V(\mathcal{B}) \cap V(\mathcal{C})$ implies that the sets $V(\mathcal{A}) \cap V_1(\mathcal{C})$ and $V(\mathcal{B}) \cap V_1(\mathcal{C})$ partition $V_1(\mathcal{C})$. To show that $V_1(\mathcal{A}) \cap V_1(\mathcal{C})$ and $V_1(\mathcal{B}) \cap V_1(\mathcal{C})$ partition $V_1(\mathcal{C})$, it suffices to observe that $V(\mathcal{A}) \cap V_1(\mathcal{C}) = V_1(\mathcal{A}) \cap V_1(\mathcal{C})$ and similarly for $\mathcal{B}$: note that $V(\mathcal{A}) = V_1(\mathcal{A}) \cup V_2(\mathcal{A})$ and that $V_2(\mathcal{A}) \cap V_1(\mathcal{C}) = \emptyset$ since $V_2(\mathcal{A}) \subseteq R(\mathcal{C}) \cup V_2(\mathcal{C})$.

(iii) Let $abcd$ be a P_4 of $\mathcal{C}$. According to Lemma 2.10 (statement (iv)), one of the midpoints b, c belongs to $V(\mathcal{A})$; we suppose without loss of generality that $b \in V(\mathcal{A})$. Then the general form of the P_4s of type (6) implies that b is a midpoint of $\mathcal{A}$, i.e., $b \in V_1(\mathcal{A})$. Moreover, according to Lemma 2.10 (statement (iii)), $b \in V(\mathcal{A})$ implies that $d \in V(\mathcal{A})$; in particular, $d \in V_2(\mathcal{A})$. On the other hand, $a, c \notin V(\mathcal{A})$, for otherwise no midpoint or no endpoint of the P_4 $abcd$ would belong to $V(\mathcal{B})$ given that the sets $V(\mathcal{A}) \cap V(\mathcal{C})$ and $V(\mathcal{B}) \cap V(\mathcal{C})$ are disjoint, in contradiction to Lemma 2.10 (statement (iv)). Since $a \notin V(\mathcal{A})$ and a is adjacent to the vertex b and not adjacent to the vertex d of $\mathcal{A}$, then $a \in R(\mathcal{A})$. On the other hand, since $c \notin V(\mathcal{A})$ and c is adjacent to both the midpoint b and the endpoint d of $\mathcal{A}$, then $c \in P(\mathcal{A})$. Therefore, the P_4 $abcd$ is of type (6) with respect to the P_4-component $\mathcal{A}$. Since this holds for any P_4 of $\mathcal{C}$, the P_4-component $\mathcal{C}$ is of type B with respect to $\mathcal{A}$. By symmetry, $\mathcal{C}$ is of type B with respect to $\mathcal{B}$.

(iv) Let $xyzw$ be a P_4 of $\mathcal{A}$ and suppose without loss of generality that $y \in V(\mathcal{C})$; then $x \in R(\mathcal{C})$, $y \in V_1(\mathcal{C})$, $z \in P(\mathcal{C})$, and $w \in V_2(\mathcal{C})$. Thus, $y, w \in V(\mathcal{A}) \cap V(\mathcal{C})$. Then statement (ii) implies that $y, w \notin V(\mathcal{B})$. On the other hand, since y is a midpoint of $\mathcal{C}$, there exists a P_4, say, $aycd$, of $\mathcal{C}$, where $c \in V_1(\mathcal{C})$ and $a, d \in V_2(\mathcal{C})$. Then the path $xcza$ is a P_4 and belongs to $\mathcal{B}$ (note that the edge xc belongs to $\mathcal{B}$), which implies that $x \in V_2(\mathcal{B})$ and $z \in V_1(\mathcal{B})$. Since $y \notin V(\mathcal{B})$, and y is adjacent to both the endpoint x and the midpoint z of $\mathcal{B}$, then $y \in P(\mathcal{B})$. Moreover, since $w \notin V(\mathcal{B})$, and w is adjacent to z and not adjacent to x, then $w \in R(\mathcal{B})$. Therefore, the P_4 $xyzw$ is of type (6) with respect to the P_4-component $\mathcal{B}$. Since the P_4 $xyzw$ is an arbitrary P_4 of the P_4-component $\mathcal{A}$, $\mathcal{A}$ is of type B with respect to $\mathcal{B}$. By symmetry, $\mathcal{B}$ is of type B with respect to $\mathcal{A}$. □

Note that statement (ii) of Lemma 2.11 implies that, for a P_4-component $\mathcal{C}$ meeting the conditions of the lemma, the subgraph spanned by the ribs of the P_4s in $\mathcal{C}$ is bipartite. It is worth mentioning here that the subgraph spanned by the ribs of the P_4s in a separable P_4-component may very well not be bipartite. For example, the ribs of the P_4s in the P_4-component of the graph of Figure 2 induced by the vertex set $\{a_1, a_2, b, c_1, c_2, d_1, d_2\}$ span a triangle, i.e., a graph which is not bipartite. If $\mathcal{C}$ is such a P_4-component (i.e., $\mathcal{C}$ is a separable P_4-component such that the subgraph spanned by the ribs of its P_4s is not bipartite) and if $\mathcal{B}$ is a P_4-component which is of type B with respect to $\mathcal{C}$, then $V(\mathcal{C}) \subset V(\mathcal{B})$; see Figure 2 for an example and the Appendix for a proof.

LEMMA 2.12. *Let $\mathcal{A}, \mathcal{B}$, and $\mathcal{C}$ be three distinct non-trivial P_4-components of a graph G such that $\mathcal{A}$ is of type B with respect to $\mathcal{B}$, $\mathcal{B}$ is of type B with respect to $\mathcal{C}$ and $|V(\mathcal{A})| \geq |V(\mathcal{C})|$. Then, if there exists a vertex which is a midpoint of all three components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, the P_4-component $\mathcal{A}$ is of type B with respect to $\mathcal{C}$.*

PROOF. The conditions in the statement of the lemma and Lemma 2.10 (statement (i)) imply that all three P_4-components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ are separable, and therefore their sets of midpoints and endpoints are well defined. Below, for a separable P_4-component $\mathcal{K}$, the sets $V_1(\mathcal{K})$ and $V_2(\mathcal{K})$ denote the sets of midpoints and endpoints of the P_4s of $\mathcal{K}$, and the sets $R(\mathcal{K})$ and $P(\mathcal{K})$ the partition sets of the vertices of $V(G) - V(\mathcal{K})$ as described earlier.

Let b be the vertex which is a midpoint of all three components. Since b is a midpoint of $\mathcal{A}$, there exists a P_4, say, $abcd$, of $\mathcal{A}$ with b as a midpoint. Because $\mathcal{A}$ is of type B

with respect to $\mathcal{B}$, the P_4 $abcd$ is of type (6) with respect to $\mathcal{B}$; in particular, $a \in R(\mathcal{B})$, $b \in V_1(\mathcal{B})$, $c \in P(\mathcal{B})$, and $d \in V_2(\mathcal{B})$. Since $d \in V_2(\mathcal{B})$ and given that $\mathcal{B}$ is of type B with respect to $\mathcal{C}$, then either $d \in R(\mathcal{C})$ or $d \in V_2(\mathcal{C})$. The former is not possible, since $b \in V_1(\mathcal{C})$ and b and d are not adjacent in G (recall that the path $abcd$ is a P_4). Therefore, $d \in V_2(\mathcal{C})$. On the other hand, $a \notin V(\mathcal{C})$. Otherwise, both endpoints of the edge ab, which belongs to $\mathcal{A}$, would belong to $V(\mathcal{C})$; then, according to Lemma 2.7, $V(\mathcal{A}) \subseteq V(\mathcal{C})$. Since $|V(\mathcal{A})| \geq |V(\mathcal{C})|$, we have that $V(\mathcal{A}) = V(\mathcal{C})$, and then Lemma 2.6 would imply that $\mathcal{A} = \mathcal{C}$, a contradiction, as the three P_4-components are distinct. Since $a \notin V(\mathcal{C})$ and given that a is adjacent to the midpoint b and not adjacent to the endpoint d of $\mathcal{C}$, we conclude that $a \in R(\mathcal{C})$. Finally, in a fashion similar to the one that we used for a, we can show that $c \notin V(\mathcal{C})$. Since c is adjacent to both the midpoint b and the endpoint d of $\mathcal{C}$, we conclude that $c \in P(\mathcal{C})$. Therefore, the P_4 $abcd$ of $\mathcal{A}$ is of type (6) with respect to $\mathcal{C}$.

In light of what we showed for the P_4 $abcd$ of the P_4-component $\mathcal{A}$ and due to Lemma 2.1 (statement (i)), establishing that all the P_4s of $\mathcal{A}$ are of type (6) with respect to $\mathcal{C}$ follows from proving that if ρ is a P_4 of $\mathcal{A}$ such that ρ is of type (6) with respect to $\mathcal{C}$ and one of ρ's midpoints is a midpoint of all three components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, then any P_4 adjacent to ρ also satisfies these conditions, that is, it is of type (6) with respect to $\mathcal{C}$ and it has a midpoint which is a midpoint of all three components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$.

We consider a P_4 $xyzw$ of $\mathcal{A}$ which is of type (6) with respect to $\mathcal{C}$ and suppose that its midpoint y is a midpoint of all three components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. Let $x'y'z'w'$ be a P_4 adjacent to $xyzw$; then $x' = x$ and $y' = y$, or $y' = y$ and $z' = z$, or $z' = z$ and $w' = w$. We consider these three cases separately:

Case (*a*): $x' = x$ *and* $y' = y$. Because $xyzw$ is a P_4 of type (6) with respect to $\mathcal{C}$, and y is a midpoint of $\mathcal{C}$, then $x \in R(\mathcal{C})$ and $y \in V_1(\mathcal{C})$, or, equivalently, $x' \in R(\mathcal{C})$ and $y' \in V_1(\mathcal{C})$ since $x' = x$ and $y' = y$. Moreover, since the P_4-component $\mathcal{A}$ is of type B with respect to $\mathcal{B}$, the P_4 $x'y'z'w'$ of $\mathcal{A}$ is of type (6) with respect to $\mathcal{B}$; then $w' \in V_2(\mathcal{B})$ due to the form of a P_4 of type (6) and the fact that y' (which coincides with y) is a midpoint of $\mathcal{B}$. Additionally, the fact that the P_4-component $\mathcal{B}$ is of type B with respect to $\mathcal{C}$ implies that the endpoint w' is an endpoint of a P_4 of type (6) with respect to $\mathcal{C}$ and thus belongs either to $R(\mathcal{C})$ or to $V_2(\mathcal{C})$. The former is not possible, since y' is a midpoint of $\mathcal{C}$ and w' is not adjacent to it; recall the P_4 $x'y'z'w'$. Therefore, $w' \in V_2(\mathcal{C})$. On the other hand, $z' \notin V(\mathcal{C})$. Otherwise, both endpoints of the edge $y'z'$ (which belongs to $\mathcal{A}$) would belong to $\mathcal{C}$, and then, according to Lemma 2.7, $V(\mathcal{A}) \subseteq V(\mathcal{C})$; since $|V(\mathcal{A})| \geq |V(\mathcal{C})|$, we would have that $V(\mathcal{A}) = V(\mathcal{C})$, which leads to a contradiction since Lemma 2.6 would imply that $\mathcal{A} = \mathcal{C}$. Because $z' \notin V(\mathcal{C})$, and z' is adjacent to both the midpoint y' and the endpoint w' of $\mathcal{C}$, we conclude that $z' \in P(\mathcal{C})$.

Case (*b*): $y' = y$ *and* $z' = z$. We work in a fashion similar to the one used in the previous case. Clearly, $y' \in V_1(\mathcal{C})$ and $z' \in P(\mathcal{C})$. As in the previous case, $w' \in V_2(\mathcal{C})$. On the other hand, $x' \notin V(\mathcal{C})$, which implies that $x' \in R(\mathcal{C})$, since x' is adjacent to the midpoint y' and not adjacent to the endpoint w' of $\mathcal{C}$.

Case (*c*): $z' = z$ *and* $w' = w$. From $z' = z$ and $w' = w$, and from the fact that the P_4 $xyzw$ of $\mathcal{A}$ is of type (6) with respect to $\mathcal{C}$, where y is a midpoint of $\mathcal{C}$, we conclude

that $z' \in P(\mathcal{C})$ and $w' \in V_2(\mathcal{C})$. Moreover, since $\mathcal{A}$ is of type B with respect to $\mathcal{B}$, and since the P_4 $xyzw$ belongs to $\mathcal{A}$, where y is a midpoint of $\mathcal{B}$, we conclude that $w \in V_2(\mathcal{B})$. Since $w' = w$, we have that $w' \in V_2(\mathcal{B})$, which along with the fact that the P_4 $x'y'z'w'$ belongs to $\mathcal{A}$ too and thus is of type (6) with respect to $\mathcal{B}$ implies that $y' \in V_1(\mathcal{B})$. In turn, because the P_4-component $\mathcal{B}$ is of type B with respect to $\mathcal{C}$, the midpoint y' is a midpoint of a P_4 of type (6) with respect to $\mathcal{C}$ and thus belongs either to $V_1(\mathcal{C})$ or to $P(\mathcal{C})$. The latter is not possible, since y' is not adjacent to the endpoint w' of $\mathcal{C}$. Therefore, $y' \in V_1(\mathcal{C})$. Finally, as in the previous case, $x' \in R(\mathcal{C})$.

In all three cases we conclude that the P_4 $x'y'z'w'$ is of type (6) with respect to $\mathcal{C}$ and that its midpoint y' is a midpoint of all three components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, as desired. □

We close this section by showing that the assignment of compatible directions in all the P_4s of a P_4-component does not imply that the component is necessarily acyclic. We first give an example of a graph that has a P_4-component with a directed cycle of length 3, and then we generalize it to P_4-components with directed cycles of arbitrary length. Consider the graph of Figure 5(a); each vertex is adjacent to all but two other vertices so that the paths $x_0y_0y_1z_0$, $x_1y_1y_2z_1$, and $x_2y_2y_0z_2$ are all P_4s. Additionally, the paths $y_1z_0z_1x_0$ and $z_1x_0x_1y_1$ are P_4s, are adjacent since they share the edge z_1x_0, and belong to the same P_4-component as $x_0y_0y_1z_0$ and $x_1y_1y_2z_1$ because they form the following sequence of adjacent P_4s: $x_0y_0y_1z_0$, $y_1z_0z_1x_0$, $z_1x_0x_1y_1$, $x_1y_1y_2z_1$. Assuming (without loss of generality) that the edge y_0y_1 is oriented towards y_1, this sequence of P_4s implies that the edge y_1y_2 is oriented towards y_2. In a similar fashion, the P_4 $x_2y_2y_0z_2$ belongs to the same P_4-component and the edge y_2y_0 is oriented towards y_0. Thus, a directed cycle of length 3 is formed. In fact, this is not the only directed cycle of length 3 in the P_4-component; two more are formed by the directed edges in $x_0x_1x_2$ and in $z_0z_1z_2$.

The previous example can be easily generalized to yield a graph with a P_4-component exhibiting an arbitrarily long directed cycle. Let k be an integer at least equal to 3, and let $X_k = \{x_i \mid 0 \leq i < k\}$, $Y_k = \{y_i \mid 0 \leq i < k\}$, and $Z_k = \{z_i \mid 0 \leq i < k\}$ be three sets

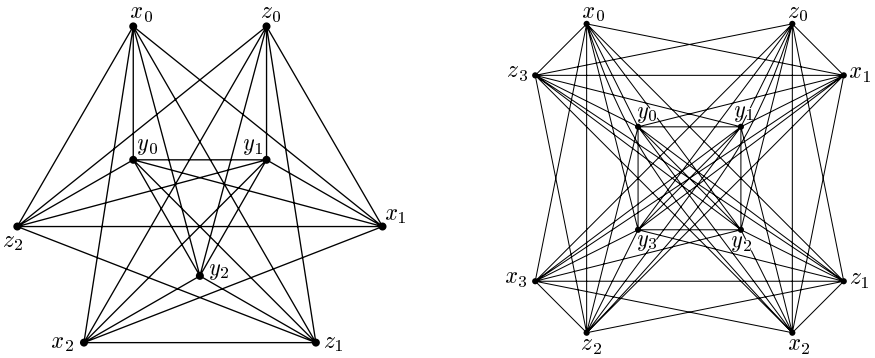

Fig. 5. Graphs that have P_4-components with cyclic P_4-transitive orientation.

of distinct vertices. We consider the graph $G_k = (V_k, E_k)$ where

$$V_k = X_k \cup Y_k \cup Z_k$$

and

$$E_k = V_k \times V_k - (\{x_i y_{i+1} \mid 0 \leq i < k\} \cup \{x_i z_i \mid 0 \leq i < k\} \cup \{y_i z_i \mid 0 \leq i < k\}).$$

The addition in the subscripts is assumed to be done mod k. Figure 5(a), (b) depicts G_3 and G_4, respectively. Then the following lemma holds.

LEMMA 2.13. *The graph G_k has the following properties*:

(i) *The only P_4s of G_k are the paths $x_i y_i y_{i+1} z_i$, $y_{i+1} z_i z_{i+1} x_i$, and $y_{i+1} x_{i+1} x_i z_{i+1}$ for $0 \leq i < k$.*
(ii) *The graph G_k has a single non-trivial P_4-component.*
(iii) *The directed edges $y_i y_{i+1}$ $(0 \leq i < k)$ form a directed cycle of length k in the non-trivial P_4-component of G_k.*
(iv) *No directed cycle of length less than k exists in the non-trivial P_4-component of G_k.*

PROOF. (i) Let $abcd$ be a P_4 of the graph G_k. First, suppose that the vertex a is y_{i+1} (for some value of $i+1$ in $\{0, \ldots, k-1\}$). Then the vertices c and d can only be x_i and z_{i+1}, since these are the only vertices of G_k not adjacent to y_{i+1}: if $d = x_i$, then $b = z_i$, and the P_4 is $y_{i+1} z_i z_{i+1} x_i$; if $d = z_{i+1}$, then $b = x_{i+1}$, and the P_4 is $y_{i+1} x_{i+1} x_i z_{i+1}$. These are the last two P_4s in the statement of the lemma. Now, suppose that $a \notin Y_k$; we may also assume without loss of generality that $d \notin Y_k$, thus avoiding getting the P_4s of the previous case again (traversed from back to front). However, then $a, d \in X_k \cup Z_k$; since a and d are not adjacent, they can only be x_i and z_i for some $i = 0, \ldots, k-1$. Moreover, the remaining two vertices b and c, which are not adjacent to d and a, respectively, can only be y_i and y_{i+1}. Therefore the P_4s in this case are the paths $x_i y_i y_{i+1} z_i$.

(ii) This property follows from the fact that the P_4s $x_i y_i y_{i+1} z_i$, $y_{i+1} z_i z_{i+1} x_i$, $z_{i+1} x_i x_{i+1} y_{i+1}$, and $x_{i+1} y_{i+1} y_{i+2} z_{i+1}$ are adjacent and therefore belong to the same P_4-component for all i such that $0 \leq i < k$.

(iii) The sequence of P_4s in the proof of property (ii) implies that if the edge $y_i y_{i+1}$ is oriented towards y_{i+1}, then the edge $y_{i+1} y_{i+2}$ will be oriented towards y_{i+2}. The property follows.

(iv) From the P_4s of the graph G_k (see property (i)), we note that all their edges connect vertices whose subscripts differ by at most 1. We assume without loss of generality that the edges $y_i y_{i+1}$ are oriented towards y_{i+1}. Then, from the P_4 $x_i y_i y_{i+1} z_i$, the edge $x_i y_i$ is oriented towards x_i and the edge $y_{i+1} z_i$ towards y_{i+1}. Since the edge $y_{i+1} z_i$ is oriented towards y_{i+1}, the edges $z_i z_{i+1}$ and $z_{i+1} x_i$ of the P_4 $y_{i+1} z_i z_{i+1} x_i$ are both oriented towards z_{i+1}. Finally, since the edge $x_i z_{i+1}$ is oriented towards z_{i+1}, the edges $y_{i+1} x_{i+1}$ and $x_{i+1} x_i$ of the P_4 $y_{i+1} x_{i+1} x_i z_{i+1}$ are both oriented towards x_{i+1}. In other words, all the edges of the form $a_i b_{i+1}$ are oriented from a_i to b_{i+1}, whereas the only edges connecting vertices with the same subscript are the edges $x_i y_i$ which are oriented towards x_i; this implies that the length of a directed cycle of the P_4-component cannot be less than k. □

3. Recognition of P_4-Comparability Graphs. Our recognition algorithm works by constructing and orienting the P_4-components of the input graph, say, G, and then by checking whether they are acyclic (Lemma 2.2). The P_4-components are constructed as follows: the algorithm considers initially m (partial) P_4-components, one for each edge of G; then it locates the P_3s in all the P_4s of G, and whenever the edges of such a P_3 belong to different (partial) P_4-components it unions and appropriately orients these P_4-components. Since we are interested in a P_4-transitive orientation of each P_4-component, the edges of such a P_3 need to be oriented either towards their common endpoint or away from it. It is important to note that the orientation of any two edges belonging to the same P_4-component is not free to change relative to each other (Lemma 2.1, statement (iii)); either the orientation of all the edges in the component stays the same or is inverted for all the edges. If no compatible orientation can be found (in which case the P_4-component does not admit a P_4-transitive orientation) or if the resulting orientation contains directed cycles, then the input graph G is not a P_4-comparability graph.

As stated earlier, the P_4s of the graph G are computed by means of processing the BFS-trees of the complement $\overline{G}$ of G rooted at each of its vertices. It is important to observe that if $abcd$ is a P_4 of G, then its complement is the P_4 $bdac$ and it belongs to the complement $\overline{G}$ of G. We consider the BFS-tree $T_{\overline{G}}(b)$ of $\overline{G}$ rooted at b. Since $bdac$ is a P_4 of $\overline{G}$, the vertices b, d, and a have to belong to the zeroth, first, and second level of $T_{\overline{G}}(b)$, respectively; the vertex c may belong to the second or third level, but not to the first level since c is not adjacent to b in $\overline{G}$. These two cases are shown in Figure 6.

Our algorithm also takes advantage of the following result in order to achieve its stated time complexity.

LEMMA 3.1. *Let G be a graph and let $T_{\overline{G}}(v)$ be the BFS-tree of the complement $\overline{G}$ rooted at a vertex v. Then the number of vertices in all the levels of $T_{\overline{G}}(v)$, except for the zeroth and the first, does not exceed the degree of v in G.*

PROOF. Clearly true, since the vertices in all the levels of $T_{\overline{G}}(v)$, except for the zeroth and the first, are vertices which are not adjacent to v in $\overline{G}$. □

The algorithm is described in more detail below. We assume that the input graph is connected; the case of disconnected graphs is addressed in Section 3.3.

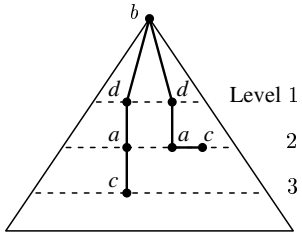

Fig. 6. The two positions of the P_4 $bdac$ in the BFS-tree $T_{\overline{G}}(b)$.

Recognition Algorithm

Input: a connected graph G on n vertices and m edges.
Output: yes, if G is a P_4-comparability graph; otherwise, no.

1. Initialize to 0 all the entries of an array $M[\,]$ which is of size n;
   for each edge e of the graph G, do
   assign to e an arbitrary orientation;
   construct a P_4-component containing only the edge e;
2. For each vertex v of the graph G, do
   2.1. compute the sets L_1, L_2, and L_3 of vertices in the first, second, and third level, respectively of the BFS-tree of the complement $\overline{G}$ rooted at v;
   2.2. partition the set L_2 into subsets of vertices so that two vertices belong to the same subset iff they have (in $\overline{G}$) the same neighbors in L_1;
   2.3. for each vertex x in L_2, do
      2.3.1. for each vertex w adjacent to x in G, do
         $M[w] \leftarrow 1$; {*mark in* $M[\,]$ *the neighbors of* x *in* G}
      2.3.2. for each vertex y in L_3 do
         if $M[y] = 0$
         then {xvy *is a* P_3 *in a* P_4 *of* G}
            If the edges xv and vy belong to the same P_4-component and do not both point towards v or away from it, then the P_4-component cannot admit a P_4-transitive orientation and we conclude that the graph G is not a P_4-comparability graph.
            If the edges xv and vy belong to different P_4-components, then we union these components into a single component and if the edges do not both point towards v or away from it, we invert (during the unioning) the orientation of all the edges of the unioned P_4-component with the fewest edges.
      2.3.3. for each vertex y in L_2 do
         if $M[y] = 0$ and the vertices x and y belong to different partition sets of L_2 (produced in Step 2.2)
         then {xvy *is a* P_3 *in a* P_4 *of* G}
            process the edges xv and vy as in Step 2.3.2;
      2.3.4. for each vertex w adjacent to x in G, do
         $M[w] \leftarrow 0$; {*clear* $M[\,]$}
3. After all the vertices have been processed, we apply topological sorting on the directed graph spanned by the directed edges associated with each of the resulting non-trivial P_4-components; if the topological sorting succeeds, then the component is acyclic, otherwise there is a directed cycle. If any of the P_4-components contains a directed cycle, then the graph is not a P_4-comparability graph.

For each P_4-component we maintain a linked list of the records of the edges in the component, and the total number of these edges. Each edge record contains a pointer to the header record of the component to which the edge belongs; in this way, we can

determine in constant time the component to which an edge belongs and the component's size. Unioning two P_4-components is done by updating the edge records of the smallest component and by linking them to the edge list of the largest one, which implies that the union operation takes time linear in the size of the smallest component. As mentioned above, in the process of unioning, we may have to invert the orientation in the edge records that we link, if the current orientations are not compatible.

Correctness of the Recognition Algorithm. The correctness of the algorithm follows (i) from the fact that in Steps 2.3.2 and 2.3.3 it processes the P_3s in all the P_4s of the input graph G (Lemmata 3.2 and 3.3) and that it assigns correct orientations on the edges of these P_3s, (ii) from the correct construction of the P_4-components by unioning partial P_4-components whenever a P_3 is processed whose edges belong to more than one such partial component, and (iii) from Lemma 2.2 in conjunction with Step 3 of the algorithm.

Note that the initial assignment of 0 to all the entries of the array $M[\,]$ and the clearing of all the set entries at Step 2.3.4 of the algorithm guarantee that the only entries of the array which are equal to 1 at any iteration are precisely those corresponding to the vertices adjacent in G to the vertex processed at Step 2.3.

LEMMA 3.2. *Every P_3 in a P_4 of the input graph G is considered at Steps* 2.3.2 *or* 2.3.3 *of the recognition algorithm.*

PROOF. Let $abcd$ be a P_4 of the graph G; we will show that the P_3 abc is considered at Step 2.3.2 or 2.3.3 of the recognition algorithm. Since the algorithm processes each vertex v of G in Step 2 and considers the BFS-tree of $\overline{G}$ rooted at v, it will process b, it will consider the BFS-tree $T_{\overline{G}}(b)$ of $\overline{G}$ rooted at b, and it will compute the sets L_1, L_2, and L_3 of vertices in the first, second, and third level of $T_{\overline{G}}(b)$, respectively. We consider the two cases of Figure 6. In the first case the vertices a and c belong to the second and third level of $T_{\overline{G}}(b)$, respectively, and they are adjacent in $\overline{G}$. Thus, $a \in L_2$ and $c \in L_3$. Moreover, since a and c are adjacent in $\overline{G}$, then a and c are not adjacent in G. Hence, $M[c] = 0$ when $x = a$ in Step 2.3. Therefore, the P_3 abc is considered in Step 2.3.2 when $x = a$ and $y = c$. In the second case of Figure 6, the vertices a and c belong to the second level of $T_{\overline{G}}(b)$, they are adjacent in $\overline{G}$, and a is adjacent to $d \in L_1$ in $\overline{G}$ whereas c is not. Thus, $a \in L_2$, $c \in L_2$, $M[c] = 0$ when $x = a$ in Step 2.3, and the vertices a and c belong to different sets in the partition of the vertices in L_2 depending on the vertices in L_1 to which they are adjacent in $\overline{G}$. Therefore, the P_3 abc is considered in Step 2.3.3 when $x = a$ and $y = c$. □

LEMMA 3.3. *The vertices x, v, y considered at Steps* 2.3.2 *and* 2.3.3 *of the recognition algorithm induce the P_3 xvy of the input graph G which participates in a P_4 of G.*

PROOF. We first consider Step 2.3.2; in this case the vertices x and y are in the second and third level of $T_{\overline{G}}(v)$, respectively. Then the path vp_xxy is a P_4 in $\overline{G}$, where p_x is the parent of x in $T_{\overline{G}}(v)$. This implies that $xvyp_x$ is a P_4 in G and xvy is a P_3 in a P_4 of G.

We now consider Step 2.3.3. Then the vertices x and y are in the second level of the BFS-tree $T_{\overline{G}}(v)$ of $\overline{G}$ rooted at v. Moreover, since $M[y] = 0$, x and y are not adjacent in G, that is, they are adjacent in $\overline{G}$. Finally, the fact that x and y do not belong to the

same partition set of L_2, implies that there is a vertex in the first level of $T_{\overline{G}}(v)$ which is adjacent to one of them in $\overline{G}$ and not to the other one. Suppose that this vertex is z and that it is adjacent to x; the case where z is adjacent to y and not to x is similar. Then the path $vzxy$ is a P_4 in $\overline{G}$, which implies that $xvyz$ is a P_4 in G. Clearly, xvy is a P_3 in a P_4 of G. □

Before analyzing the complexity of the recognition algorithm, we explain in more detail how Steps 2.1 and 2.2 are carried out.

3.1. *Computing the Vertex Sets L_1, L_2, and L_3.* The computation of these sets can be done by means of the algorithms of Dahlhaus et al. [5] and Ito and Yokoyama [15] for computing the BFS-tree of the complement of a graph in time linear in the size of the input graph; both algorithms require the construction of a special representation of the graph and rely on special cases of the disjoint-set union problem. Instead, we use another algorithm which computes the vertices in each level of the BFS-tree of the complement of a graph—i.e., it effectively implements breadth-first search on the complement—in the above stated time complexity (a similar approach is described in [6]). The algorithm is very simple and uses the standard adjacency list representation of a graph. It works by constructing each level L_{i+1} from the previous one, L_i, based on the following lemma.

LEMMA 3.4. *Let G be a graph and let L_i be the set of vertices in the i*th *level of a BFS-tree of $\overline{G}$. Consider a vertex w which does not appear in any of the levels from the* 0*th up to the k*th. *Then w is a vertex of the $(k+1)$*st *level if and only if there exists at least one vertex of L_k which is not adjacent to w in G.*

PROOF. The vertex w is a vertex of the $(k+1)$st level if and only if it is adjacent in $\overline{G}$ to at least one vertex in L_k. The lemma follows. □

We give below the description of the algorithm.

Algorithm for computing the BFS-tree of the complement of a graph G rooted at a vertex v

1. Initialize to 0 all the entries of the array $Adj[\,]$ which is of size n;
2. Construct a list L_0 containing a single record associated with the vertex v and a list S containing a record for each of the vertices of G except for v;
3. $i \leftarrow 0$;
   while the list L_i is not empty, do
   3.1. initialize the list L_{i+1} to the empty list;
   3.2. for each vertex u in L_i do
        for each vertex w adjacent to u in G do
        increment $Adj[w]$ by 1;
   3.3. for each vertex s in S do
        if $Adj[s] < |L_i|$
        then remove s from S and add it to the list L_{i+1}
        else $Adj[s] \leftarrow 0$;
   3.4. increment i by 1;

The correctness of the algorithm follows from Lemma 3.4. Note that the set S contains the vertices which, until the current iteration, have not appeared in any of the computed levels. Moreover, because of Steps 1 and 3.3, the entries of the array $Adj[\,]$ corresponding to the vertices in S are equal to 0 at the beginning of each iteration of the while loop in Step 3. In this way the test "$Adj[s] < |L_i|$" correctly tests the number of vertices of L_i which are adjacent to the vertex s in G against the size of L_i. Finally, it must be noted that when the while loop of Step 3 terminates, the list S may very well be non-empty; this happens when the graph $\overline{G}$ is disconnected.

Suppose that the input graph G has n vertices and m edges. Clearly, Steps 1 and 2 take $O(n)$ time. In each iteration of the while loop of Step 3, Steps 3.1 and 3.4 take constant time, while Step 3.2 takes $O(\sum_{u\in L_i} d_G(u))$ time, where $d_G(u)$ denotes the degree of the vertex u in G. Step 3.3 takes time linear in the current size of the list S; the elements of S can be partitioned into two sets: (i) the vertices which end up belonging to L_{i+1}, and (ii) the vertices for which the corresponding entries of the array $Adj[\,]$ are equal to $|L_i|$. The number of elements of S in the former set is equal to $|L_{i+1}|$, while the number of elements in the latter set does not exceed the sum of the degrees (in G) of the vertices in L_i. Thus, Step 3.3 takes $O\big(|L_{i+1}| + \sum_{u\in L_i} d_G(u)\big)$ time.

Therefore, the time taken by the algorithm is

$$\begin{aligned} O(n) + \sum_i \left(O(1) + O\left(|L_{i+1}| + \sum_{u\in L_i} d_G(u) \right) \right) \\ = O(n) + O\left(\sum_i (1 + |L_{i+1}|) \right) + O\Big(\sum_i \sum_{u\in L_i} d_G(u) \Big) \\ = O(n) + O(n) + O(m). \end{aligned}$$

The inequalities $\sum_i |L_i| \le n$ and $\sum_i \sum_{u\in L_i} d_G(u) \le \sum_u d_G(u) = 2m$ hold because each vertex belongs to at most one level of the BFS-tree. Moreover, the space needed is $O(n+m)$. Consequently, we have:

THEOREM 3.1. *Let G be a graph on n vertices and m edges, and let v be a vertex of G. Then the above algorithm computes the vertices in the levels of the BFS-tree of the complement $\overline{G}$ of G rooted at v in $O(n+m)$ time and $O(n+m)$ space.*

3.2. *Partitioning the Vertices in L_2.* It is not difficult to see that the partition of the vertices in L_2 depending on their neighbors in $\overline{G}$ which are in L_1 is identical to the partition of the vertices in L_2 depending on their neighbors in G which are in L_1. This is indeed so, because the subset of vertices in L_1 which are adjacent (in G) to a vertex $x \in L_2$ is $L_1 - N_x$, where N_x is the subset of L_1 containing vertices which are adjacent (in $\overline{G}$) to x. If for two vertices x and y the sets N_x and N_y are equal, then so are the sets $L_1 - N_x$ and $L_1 - N_y$, whereas if $N_x \neq N_y$ then $L_1 - N_x \neq L_1 - N_y$. Therefore, in the algorithm we work with neighbors in G, instead of working with neighbors in $\overline{G}$.

The algorithm initially considers a single set (list) which contains all the vertices of the set L_2. It then processes each vertex, say, u, of the set L_1 as follows: For each set of the current partition, we check if none, all, or only some of its elements are neighbors of u in G; in the first and second case the set is not modified, in the third case, it is split

into the subset of neighbors of u in G and the subset of non-neighbors of u in G. After all the vertices of L_1 have been processed, the resulting partition is the desired partition. The partition is stored in an array *Set*[ ] of size equal to the number of vertices of G.

Algorithm for partitioning the set L_2 in terms of adjacency to elements of the set L_1

1. Initialize to 0 the entries of the arrays $M[\,]$ and *size*[ ] which are of size n; insert all the vertices in L_2 in the list *LSet*[1] and assign *size*[1] $\leftarrow |L_2|$; $k \leftarrow 1$; {*k holds the number of sets in the partition*}
2. For each vertex u in L_1 do
   2.1. for each vertex w adjacent to u in G do
        $M[w] \leftarrow 1$; {*mark in $M[\,]$ the neighbors of u in G*}
   2.2. $k_0 \leftarrow k$;
        for each list *LSet*[i], $i = 1, 2, \ldots, k_0$, do
        2.2.1. traverse the list *LSet*[i] and count the number of its vertices which are neighbors of u in G (use the array $M[\,]$); let ℓ be the number of these vertices;
        2.2.2. if $\ell > 0$ and $\ell <$ *size*[i]
               then {*split LSet*[i]; *create a new set*}
                    increment k by 1;
                    traverse the list *LSet*[i] and for each of its vertices w which is a neighbor of u in G (use $M[\,]$), delete w from *LSet*[i] and insert it in *LSet*[k];
                    size[k] $\leftarrow \ell$;
                    decrease *size*[i] by ℓ;
   2.3. for each vertex w adjacent to u in G do
        $M[w] \leftarrow 0$; {*clear $M[\,]$*}
3. For each list *LSet*[i], $i = 1, 2, \ldots, k$, do
   traverse the list *LSet*[i] and for each of its vertices w set the entry *Set*[w] equal to i;

Note that, thanks to the array *Set*[ ], checking whether two vertices x and y belong to the same partition set of L_2 reduces to testing whether the entries *Set*[x] and *Set*[y] are equal.

The correctness of the algorithm follows from induction on the number of the processed vertices in L_1. At the basis step, when no vertices from the set L_1 have been processed, all the elements of the set L_2 belong to the same set, as desired. Suppose that after processing $i \geq 0$ vertices from L_1, the resulting partition of L_2 is correct with respect to the processed vertices. We consider the processing of the next vertex, say, u, from L_1: then only the sets which contain at least one vertex which is adjacent (in G) to u and at least one vertex which is not adjacent (in G) to u should be split, and indeed these are the only ones that are split; the splitting produces a subset of neighbors of u in G and a subset of non-neighbors of u (Step 2.2.2). Note that because of Steps 1, 2.1, and 2.3, the array $M[\,]$ is clear at the beginning of each iteration of the for loop in Step 2, so that in Step 2.2 the marked entries are precisely those corresponding to the neighbors of the current vertex u in G.

Step 1 of the algorithm clearly takes $O(n)$ time. Steps 2.1 and 2.3 take $O(d_G(u))$ time, where $d_G(u)$ is equal to the degree of u in G. Step 2.2.1 takes $O(|LSet(i)|)$ time and so does Step 2.2.2, since deleting an entry from and inserting an entry in a list can be done in constant time, and the remaining operations take constant time. Therefore, Step 2.2 takes time linear in the total size of lists $LSet[i]$ which existed when u started being processed; since the lists contained the vertices in L_2 and none of these lists was empty, we conclude that Step 2.2 takes $O(|L_2|)$ time. Then Step 2 can be executed in $O(\sum_u (d_G(u) + |L_2|)) = O(m + n|L_2|)$ time. Step 3 takes time linear in the total size of the final lists $LSet[i]$, i.e., $O(|L_2|)$ time. Thus the entire partitioning algorithm takes $O(m + n|L_2|)$ time. Since all the initialized lists $LSet[i]$ contain at least one vertex from L_2 and since these lists do not share vertices, then the space complexity is $O(n + |L_2|) = O(n)$.

The results of the paragraph are summarized in the following theorem.

THEOREM 3.2. *Let G be a graph on n vertices and m edges, and let L_1 and L_2 be two disjoint sets of vertices. Then the above algorithm partitions the vertices in L_2 depending on their neighbors in $\overline{G}$ which belong to L_1 in $O(m + n|L_2|)$ time and $O(n)$ space.*

Time and Space Complexity of the Recognition Algorithm. Clearly, Step 1 of the algorithm takes $O(n + m)$ time. Steps 2.1 and 2.2 while processing each one of the vertices of G take $O(n+m) = O(m)$ and $O(m+n|L_2|)$ time, respectively (in accordance with Theorems 3.1 and 3.2), and Steps 2.3.1 and 2.3.4 take $O(d_G(x))$ time where $d_G(x)$ denotes the degree of vertex x in G. If we ignore the cost of unioning P_4-components, then Steps 2.3.2 and 2.3.3 require $O(1)$ time per vertex in L_3 and L_2, respectively; recall that testing whether two vertices belong to the same partition set of L_2 takes constant time. If we take into account Lemma 3.1, we have that $|L_2| \leq d_G(v)$ and $|L_3| \leq d_G(v)$. Therefore, provided that P_4-component unioning is ignored, the time complexity of Step 2 of the algorithm is

$$T_2 = \sum_v \left(O(m + n\, d_G(v)) + \sum_x O(d_G(v) + d_G(x)) \right).$$

By observing that x belongs to L_2, we conclude that x assumes at most $d_G(v)$ different values. Thus,

$$\begin{aligned} T_2 &= O\left(\sum_v m + n \sum_v d_G(v)\right) + O\left(\sum_v \sum_x (d_G(v) + d_G(x))\right) \\ &= O(nm) + O(nm) + O\left(\sum_v \left(d_G^2(v) + \sum_x d_G(x)\right)\right) \\ &= O(nm) + O\left(\sum_v d_G^2(v)\right) + O\left(\sum_v \sum_x d_G(x)\right) \\ &= O(nm) \end{aligned}$$

since $\sum_v d_G^2(v) \leq n \sum_v d_G(v) = O(nm)$ and $\sum_v \sum_x d_G(x) \leq \sum_v 2m = 2nm$. Now, the time required for all the P_4-component union operations during the processing of all

the vertices is $O(m \log m)$ [1]; there cannot be more than $m - 1$ such operations (we start with m P_4-components and we may end up with only one), and each one of them takes time linear in the size of the smallest of the two components that are unioned.

Finally, constructing the directed graph from the edges associated with a non-trivial P_4-component and checking whether it is acyclic takes $O(n+m_i)$, where m_i is the number of edges of the component. Thus, the total time taken by Step 2 is $O\left(\sum_i (n + m_i)\right) = O(nm)$, since there are at most m P_4-components and $\sum_i m_i = m$. Thus, the overall time complexity is $O(n+nm+m \log m+nm) = O(nm)$; note that $\log m \leq 2 \log n = O(n)$.

The space complexity is linear in the size of the graph G: the arrays $M[\,]$ and $Set[\,]$ take linear space, both Steps 2.1 and 2.2 require linear space (Theorems 3.1 and 3.2), the set L_1 is represented as a list of $O(n)$ size, the sets L_2 and L_3 are represented as lists having $O(d_G(v))$ size each, and the handling of the P_4-components requires one record per edge and one record per component. Thus, the space required is $O(n + m)$.

Therefore, we have the following result:

THEOREM 3.3. *It can be decided whether a connected graph on n vertices and m edges is a P_4-comparability graph in $O(nm)$ time and $O(n + m)$ space.*

3.3. *The Case of Disconnected Input Graphs.* If the input graph is disconnected, we compute its connected components and work on each one of them as indicated above. In light of Theorem 3.3 and since the connected components of a graph can be computed in time and space linear in the size of the graph by means of depth-first search [1], we conclude that the overall time complexity is $O(n+m)+\sum_i O(n_i m_i) = O(n \sum m_i) = O(nm)$ and the space is $O(n+m)+\sum_i O(n_i+m_i) = O(n+m)$ since $\sum_i n_i = n$ and $\sum_i m_i = m$.

THEOREM 3.4. *It can be decided whether a graph on n vertices and m edges is a P_4-comparability graph in $O(nm)$ time and $O(n + m)$ space.*

4. Acyclic P_4-Transitive Orientation. Although each of the P_4-components of the input graph produced by the recognition algorithm is acyclic, directed cycles may arise when all the P_4-components are placed together; obviously, these cycles will include edges from more than one P_4-component. Inversion of the orientations of some of the components will yield the desired acyclic P_4-transitive orientation; Lemma 4.3 provides the rule for fixing the P_4-transitive orientations of the non-trivial P_4-components, so that the resulting overall orientation of the input graph is acyclic. Before that, we need two additional useful results.

LEMMA 4.1. *Let $\mathcal{B}$, $\mathcal{C}$ be two non-trivial P_4-components of a graph such that $\mathcal{B}$ is of type B with respect to $\mathcal{C}$. Then, in a P_4-transitive orientation of $\mathcal{C}$ (assuming that $\mathcal{C}$ admits such an orientation),*

(i) *if an edge of $\mathcal{B}$ is oriented towards its endpoint that belongs to $V(\mathcal{C})$, then so do all the edges of $\mathcal{B}$;*
(ii) *the edges of $\mathcal{B}$ incident upon the same vertex v are all oriented either towards v or away from it.*

PROOF. (i) Since $\mathcal{B}$ is of type B with respect to $\mathcal{C}$, any P_4 $abcd$ of $\mathcal{B}$ is of type (6) with respect to $\mathcal{C}$. The general form of such P_4s implies that $a \in R(\mathcal{C})$, $b \in V_1(\mathcal{C})$, $c \in P(\mathcal{C})$, and $d \in V_2(\mathcal{C})$ (see Section 2). The truth of the statement follows from the fact that any P_4 of $\mathcal{B}$ has exactly two vertices belonging to $V(\mathcal{C})$ which are at distance 2 apart, and the fact that the sets $R(\mathcal{C})$, $V_1(\mathcal{C})$, $P(\mathcal{C})$, $V_2(\mathcal{C})$ are disjoint and hence two adjacent P_4s of $\mathcal{B}$ share an edge that is a rib or a wing to both of them.

(ii) Follows easily from statement (i): if $v \in V(\mathcal{C})$, then all the edges of $\mathcal{B}$ incident upon v are oriented towards v; otherwise, they are oriented away from v. □

LEMMA 4.2. *Suppose that the non-trivial P_4-components of a graph G have received acyclic P_4-transitive orientations. If the directed subgraph of G spanned by the edges of these P_4-components contains a directed triangle, then the three edges of the triangle belong to three different P_4-components which are of type B with respect to one another.*

PROOF. Let a, b, c be the vertices of the directed triangle, and suppose that the edges ab, ac, and bc belong to the P_4-components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, respectively; it is not assumed that the three P_4-components are distinct. We suppose without loss of generality that $|V(\mathcal{C})| \leq |V(\mathcal{A})|$ and $|V(\mathcal{C})| \leq |V(\mathcal{B}|$. Then the vertex a does not belong to $V(\mathcal{C})$. If $a \in V(\mathcal{C})$, then the edge ab which belongs to $\mathcal{A}$ would have both endpoints in $V(\mathcal{C})$. This would imply that $V(\mathcal{A}) \subseteq V(\mathcal{C})$, according to Lemma 2.7. Moreover, since $|V(\mathcal{C})| \leq |V(\mathcal{A})|$, we have that $V(\mathcal{A}) = V(\mathcal{C})$, which implies that $\mathcal{A} = \mathcal{C}$ (Lemma 2.6). Similarly, if $a \in V(\mathcal{C})$, then from the edge ac we conclude that $\mathcal{B} = \mathcal{C}$. However, then all three edges of the directed triangle belong to the same P_4-component, in contradiction to the fact that the P_4-components of the graph G have received acyclic P_4-transitive orientations. Thus, $a \notin V(\mathcal{C})$.

Since the orientation of each P_4-component is acyclic, at least two of the P_4-components $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ must be different. In fact, they are all different. Note that if the three edges of the triangle participated in two distinct P_4-components, then $\mathcal{A} = \mathcal{B}$, since $\mathcal{C}$ cannot be identical to either $\mathcal{A}$ or $\mathcal{B}$ because $a \notin V(\mathcal{C})$. Then the edges ab and ac belong to the same P_4-component which is of type A or of type B with respect to $\mathcal{C}$, since $a \notin V(\mathcal{C})$ and $b, c \in V(\mathcal{C})$. In the former case, Lemma 2.8 would imply that the edges ab and ac would be oriented either both towards a or both away from it, and thus the triangle with vertices a, b, and c could not form a directed cycle. In the latter case, the edges ab and ac would again be oriented either both towards a or both away from it (Lemma 4.1, statement (ii)), and thus the triangle could not form a directed cycle in this case either. Therefore, the three edges of the triangle belong to three distinct P_4-components.

We consider the P_4-component $\mathcal{C}$. Because $a \notin V(\mathcal{C})$ while $b, c \in V(\mathcal{C})$, the other two components $\mathcal{A}$ and $\mathcal{B}$ are of type A or of type B with respect to $\mathcal{C}$. If any one of them were of type A, then, according to Lemma 2.8, the edges ab and ac would belong to the same P_4-component, in contradiction to the fact that $\mathcal{A} \neq \mathcal{B}$. Therefore, both $\mathcal{A}$ and $\mathcal{B}$ are of type B with respect to $\mathcal{C}$. Then, since $a \in (V(\mathcal{A}) \cap V(\mathcal{B})) - V(\mathcal{C})$, Lemma 2.11 applies implying that the three P_4-components are of type B with respect to one another. □

Lemma 4.2 identified how a directed triangle can occur when placing the P_4-transitively oriented P_4-components together. The next lemma takes advantage of this result to suggest a way to avoid directed cycles.

LEMMA 4.3. *Let $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_h$ be the non-trivial P_4-components of a graph G ordered by non-decreasing vertex number and suppose that each component has received an acyclic P_4-transitive orientation. Consider the sets $S_i = \{\mathcal{C}_j \mid j < i$ and $\mathcal{C}_i$ is of type B with respect to $\mathcal{C}_j\}$, for $i = 1, 2, \ldots, h$. If the edges of each P_4-component $\mathcal{C}_i$ for which $S_i \neq \emptyset$ get oriented towards their endpoint which belongs to $V(\mathcal{C}_{\hat{\imath}})$, where $\hat{\imath} = \min\{j \mid \mathcal{C}_j \in S_i\}$, then the directed subgraph of G spanned by the edges of the $\mathcal{C}_i$s $(1 \leq i \leq h)$ does not contain a directed cycle.*

PROOF. We suppose for contradiction that the described orientation scheme produces a directed graph that has a directed cycle. Then, in light of Lemma 2.3, there will exist an oriented triangle which forms a directed cycle. Then, by Lemma 4.2, the edges of the triangle belong to three distinct non-trivial P_4-components which are of type B with respect to one another. Let the triangle have vertices v, u, and w, and suppose that the edges uw, vw, and uv belong to the P_4-components $\mathcal{C}_j, \mathcal{C}_k$, and $\mathcal{C}_\ell$, respectively; moreover, we assume without loss of generality that $\ell = \min\{j, k, \ell\}$. Let $\hat{\jmath} = \min\{i \mid i < j$ and $\mathcal{C}_j$ is of type B with respect to $\mathcal{C}_i\}$ and $\hat{k} = \min\{i \mid i < k$ and $\mathcal{C}_k$ is of type B with respect to $\mathcal{C}_i\}$; note that $\hat{\jmath}$ and $\hat{k}$ are well defined and do not exceed ℓ, since $\ell < j$, $\ell < k$ and both $\mathcal{C}_j$ and $\mathcal{C}_k$ are of type B with respect to $\mathcal{C}_\ell$. Then, according to the statement of the lemma, the orientation convention implies that the edges of the P_4-components $\mathcal{C}_j$ and $\mathcal{C}_k$ are oriented towards their endpoint which belongs to $V(\mathcal{C}_{\hat{\jmath}})$ and $V(\mathcal{C}_{\hat{k}})$, respectively. Then $\hat{\jmath} \neq \hat{k}$; if $\hat{\jmath} = \hat{k}$, the triangle with vertices u, v, and w could not form a directed cycle, since, according to the orientation convention, the edges uw and vw, which belong to $\mathcal{C}_j$ and $\mathcal{C}_k$, respectively, would be oriented both towards w if $w \in V(\mathcal{C}_{\hat{\jmath}})$, or both away from w if $w \notin V(\mathcal{C}_{\hat{\jmath}})$. Since $\hat{\jmath} \neq \hat{k}$, we may assume without loss of generality that $\hat{\jmath} < \hat{k}$. Then $\hat{\jmath} < \ell$, since $\hat{\jmath} < \hat{k}$ and $\hat{k} \leq \ell$. We distinguish two cases:

Case (a): the P_4-component $\mathcal{C}_\ell$ is not of type B with respect to any component $\mathcal{C}_i$ for $1 \leq i < \ell$. If the P_4-components $\mathcal{C}_\ell$, $\mathcal{C}_j$, and $\mathcal{C}_{\hat{\jmath}}$ have a common midpoint, then Lemma 2.12 applies: note that $\mathcal{C}_\ell$ is of type B with respect to $\mathcal{C}_j$, $\mathcal{C}_j$ is of type B with respect to $\mathcal{C}_{\hat{\jmath}}$, and $|V(\mathcal{C}_\ell)| \geq |V(\mathcal{C}_{\hat{\jmath}})|$ since $\ell > \hat{\jmath}$. Lemma 2.12 implies that the component $\mathcal{C}_\ell$ is of type B with respect to $\mathcal{C}_{\hat{\jmath}}$, which contradicts the fact that $\mathcal{C}_\ell$ is not of type B with respect to any component $\mathcal{C}_i$ $(1 \leq i < \ell)$. If the P_4-components $\mathcal{C}_\ell$, $\mathcal{C}_j$, and $\mathcal{C}_{\hat{\jmath}}$ do not have a common midpoint, then the P_4-components $\mathcal{C}_k$, $\mathcal{C}_j$, and $\mathcal{C}_{\hat{\jmath}}$ do. Suppose for contradiction that they do not, i.e., $V_1(\mathcal{C}_k) \cap V_1(\mathcal{C}_j) \cap V_1(\mathcal{C}_{\hat{\jmath}}) = \emptyset$, where by $V_1(\mathcal{K})$ we denote the set of midpoints of a separable P_4-component $\mathcal{K}$. Moreover, from the assumption that the P_4-components $\mathcal{C}_\ell$, $\mathcal{C}_j$, and $\mathcal{C}_{\hat{\jmath}}$ do not have a common midpoint, we have that $V_1(\mathcal{C}_\ell) \cap V_1(\mathcal{C}_j) \cap V_1(\mathcal{C}_{\hat{\jmath}}) = \emptyset$. Therefore, by taking the union of these two set intersections, we find that

$$(V_1(\mathcal{C}_k) \cap V_1(\mathcal{C}_j) \cap V_1(\mathcal{C}_{\hat{\jmath}})) \cup (V_1(\mathcal{C}_\ell) \cap V_1(\mathcal{C}_j) \cap V_1(\mathcal{C}_{\hat{\jmath}})) = \emptyset$$
$$\iff \quad ((V_1(\mathcal{C}_k) \cap V_1(\mathcal{C}_j)) \cup (V_1(\mathcal{C}_\ell) \cap V_1(\mathcal{C}_j))) \cap V_1(\mathcal{C}_{\hat{\jmath}}) = \emptyset.$$

Since the P_4-components $\mathcal{C}_j$, $\mathcal{C}_k$, and $\mathcal{C}_\ell$ are of type B with respect to one another and $v \in (V(\mathcal{C}_k) \cap V(\mathcal{C}_\ell)) - V(\mathcal{C}_j)$, Lemma 2.11 (statement (ii)) implies that the sets $V_1(\mathcal{C}_k) \cap V_1(\mathcal{C}_j)$ and $V_1(\mathcal{C}_\ell) \cap V_1(\mathcal{C}_j)$ partition the set $V_1(\mathcal{C}_j)$ of midpoints of $\mathcal{C}_j$; that is,

$$(V_1(\mathcal{C}_k) \cap V_1(\mathcal{C}_j)) \cup (V_1(\mathcal{C}_\ell) \cap V_1(\mathcal{C}_j)) = V_1(\mathcal{C}_j).$$

Thus, the previous equality is equivalent to $V_1(\mathcal{C}_j) \cap V_1(\mathcal{C}_{\hat{\jmath}}) = \emptyset$. However, this comes into contradiction with the fact that $\mathcal{C}_j$ is of type B with respect to $\mathcal{C}_{\hat{\jmath}}$; therefore, the P_4-components $\mathcal{C}_k$, $\mathcal{C}_j$, and $\mathcal{C}_{\hat{\jmath}}$ have a common midpoint. Then Lemma 2.12 applies again, for the P_4-components $\mathcal{C}_k$, $\mathcal{C}_j$, and $\mathcal{C}_{\hat{\jmath}}$ this time ($\mathcal{C}_k$ is of type B with respect to $\mathcal{C}_j$, $\mathcal{C}_j$ is of type B with respect to $\mathcal{C}_{\hat{\jmath}}$, and $|V(\mathcal{C}_k)| \geq |V(\mathcal{C}_{\hat{\jmath}})|$ since $k > \hat{k} > \hat{\jmath}$), and implies that the component $\mathcal{C}_k$ is of type B with respect to $\mathcal{C}_{\hat{\jmath}}$, which contradicts the minimality of $\hat{k}$, since $\hat{\jmath} < \hat{k}$.

Case (*b*): *the P_4-component $\mathcal{C}_\ell$ is of type B with respect to a component $\mathcal{C}_i$, where* $1 \leq i < \ell$. Let $\hat{\ell} = \min\{i \mid i < \ell \text{ and } \mathcal{C}_\ell \text{ is of type B with respect to } \mathcal{C}_i\}$. If $\hat{\jmath} < \hat{\ell}$, then we reach a contradiction as in Case (a); note that $\hat{\jmath} < \hat{\ell}$ and recall that the P_4-component $\mathcal{C}_\ell$ cannot be of type B with respect to $\mathcal{C}_{\hat{\jmath}}$ as this would contradict the minimality of $\hat{\ell}$. If $\hat{\jmath} = \hat{\ell}$, then the triangle with vertices u, v, and w cannot form a directed cycle; the edges uw and uv, which belong to $\mathcal{C}_j$ and $\mathcal{C}_\ell$, respectively, get oriented both towards u if $u \in V(\mathcal{C}_{\hat{\jmath}})$, or both away from u if $u \notin V(\mathcal{C}_{\hat{\jmath}})$, according to the orientation convention in the statement of the lemma. Suppose now that $\hat{\ell} < \hat{\jmath}$. If the P_4-components $\mathcal{C}_j$, $\mathcal{C}_\ell$, and $\mathcal{C}_{\hat{\ell}}$ have a common midpoint, then Lemma 2.12 applies: note that $\mathcal{C}_j$ is of type B with respect to $\mathcal{C}_\ell$, $\mathcal{C}_\ell$ is of type B with respect to $\mathcal{C}_{\hat{\ell}}$, and $|V(\mathcal{C}_j)| \geq |V(\mathcal{C}_{\hat{\ell}})|$ since $j > \hat{\jmath} > \hat{\ell}$. Lemma 2.12 implies that the component $\mathcal{C}_j$ is of type B with respect to $\mathcal{C}_{\hat{\ell}}$, which contradicts the minimality of $\hat{\jmath}$, since $\hat{\ell} < \hat{\jmath}$. If the P_4-components $\mathcal{C}_j$, $\mathcal{C}_\ell$, and $\mathcal{C}_{\hat{\ell}}$ do not have a common midpoint, then, as in Case (a), the P_4-components $\mathcal{C}_k$, $\mathcal{C}_\ell$, and $\mathcal{C}_{\hat{\ell}}$ do. Then again Lemma 2.12 applies, implying that the component $\mathcal{C}_k$ is of type B with respect to $\mathcal{C}_{\hat{\ell}}$, which contradicts the minimality of $\hat{k}$, since $\hat{\ell} < \hat{\jmath} < \hat{k}$.

In either case we reached a contradiction, which proves that if the orientation convention described in the statement of the lemma is followed, then no directed cycle exists in the directed subgraph of G spanned by the edges of the non-trivial P_4-components of G. □

Our algorithm for computing an acyclic P_4-transitive orientation of a P_4-comparability graph employs Lemma 4.3; it processes the P_4-components of the input graph and assigns orientations in a greedy fashion by focusing on the graph edges incident upon the vertices of the non-trivial P_4-component that is currently being processed. More specifically, the algorithm works as follows:

Orientation Algorithm

1. We apply the recognition algorithm of the previous section on the input graph G, which produces the P_4-components of G and an acyclic P_4-transitive orientation of each component.
2. We sort the non-trivial P_4-components of G by (non-decreasing) number of vertices; let $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_h$ be the resulting ordered sequence. We associate with each $\mathcal{C}_i$ a `mark` and a `counter` field which are initialized to 0.

3. For each P_4-component $\mathcal{C}_i$ $(1 \leq i < h)$ in order, we do:
   By going through the vertices in $V(\mathcal{C}_i)$, we collect the edges which are incident upon a vertex in $V(\mathcal{C}_i)$ and belong to a P_4-component $\mathcal{C}_j$ where $j > i$. Then, for each such edge e, we increment the `counter` field associated with the P_4-component to which e belongs. Next, we go through the collected edges once more. This time, for each such edge e, we check whether the P_4-component to which e belongs has its `mark` field equal to 0 and its `counter` field equal to the total number of edges of the component; if yes, then this P_4-component is of type B with respect to $\mathcal{C}_i$ whereas it is not of type B with respect to any $\mathcal{C}_k$ for $k < i$, and its edges should get oriented towards their endpoint belonging to $V(\mathcal{C}_i)$; thus, in case e is not oriented towards its endpoint in $V(\mathcal{C}_i)$, we flip the component's orientation (by updating a corresponding boolean variable), and we set the `mark` field of the component to 1 to indicate that the component has received its final orientation. Finally, we set the `counter` field of the component to 0; in this way, the `counter` fields of all the non-trivial P_4-components are equal to 0 every time a P_4-component starts getting processed in Step 3.
4. We orient the edges which belong to the trivial P_4-components: this can be easily done by topologically sorting the vertices of G using only the oriented edges of the non-trivial components, and orienting the remaining edges in accordance with the topological order of their incident vertices.

Note that in Step 3 we process all the non-trivial P_4-components of the input graph G except for the largest one. This implies that the vertex set $V(\mathcal{C}_i)$ of each P_4-component $\mathcal{C}_i$ $(1 \leq i < h)$ that we process is a proper subset of the vertex set $V(G)$ of G; if $V(\mathcal{C}_i) = V(G)$, then $V(\mathcal{C}_h) = V(G)$ as well, which implies that $\mathcal{C}_i = \mathcal{C}_h$ (Lemma 2.6), a contradiction. Thus, the discussion in Section 2 regarding the P_4-components of type A and type B applies to each such $\mathcal{C}_i$. Moreover, according to Lemma 2.9, the P_4-components whose `mark` field is set to 1 in Step 3 are components which are of type B with respect to the currently processed component $\mathcal{C}_i$. Each edge of these components has exactly one endpoint in $V(\mathcal{C}_i)$ (see Lemma 2.10, statement (ii)), so that it is valid to try to orient such an edge towards that endpoint. Furthermore, Lemma 4.1 (statement (i)) implies that if such an edge gets oriented towards its endpoint which belongs to $V(\mathcal{C}_i)$, then so do all the edges of the same P_4-component. In the case that the set R in the partition of the vertices in $V(G) - V(\mathcal{C}_i)$ (as described in Section 2) is empty, there are no P_4-components of type B with respect to $\mathcal{C}_i$. While processing $\mathcal{C}_i$, our algorithm updates the `counter` fields of the components that contain an edge incident upon a vertex in $V(\mathcal{C}_i)$, finds that none of these components ends up having its `counter` field equal to the number of its edges, and thus does nothing further.

The orientation algorithm does not compute the sets R, P, and Q with respect to the currently processed P_4-component $\mathcal{C}_i$. Yet, these sets can be computed in $O(n)$ time for each $\mathcal{C}_i$ as follows. We use an array with one entry per vertex of the graph G; we initialize the array entries corresponding to vertices in $V(\mathcal{C}_i)$ to 0 and all the remaining ones to -1. Let v_1 and v_2 be an arbitrary midpoint and an arbitrary endpoint of a P_4 in $\mathcal{C}_i$. We go through the vertices adjacent to v_1 and if the vertex does not belong to $V(\mathcal{C}_i)$,

we set the corresponding entry to 1. Next, we go through the vertices adjacent to v_2; this time if the vertex does not belong to $V(\mathcal{C}_i)$, we set the corresponding entry to 2. Then the vertices in $\mathcal{C}_i$, R, P, and Q are the vertices whose corresponding array entries are equal to 0, 1, 2, and -1, respectively.

Correctness of the Orientation Algorithm. The acyclicity of the directed graph produced by our orientation algorithm relies on the following lemma.

LEMMA 4.4. *Let $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_h$ be the sequence of the non-trivial P_4-components of the input graph ordered by non-decreasing vertex number. Consider the set $S_i = \{\mathcal{C}_j \mid j < i$ and $\mathcal{C}_i$ is of type B with respect to $\mathcal{C}_j\}$ and suppose that $S_i \neq \emptyset$. If $\hat{\imath} = \min\{j \mid \mathcal{C}_j \in S_i\}$, then our algorithm orients the edges of the component $\mathcal{C}_i$ towards their endpoint which belongs to $V(\mathcal{C}_{\hat{\imath}})$.*

PROOF. The P_4-component $\mathcal{C}_i$ receives an arbitrary P_4-transitive orientation in Step 1 of the orientation algorithm. Since $\hat{\imath} = \min\{j \mid \mathcal{C}_j \in S_i\}$, then the P_4-component $\mathcal{C}_i$ is not of type B with respect to any of the components $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{\hat{\imath}-1}$; thus, its `mark` field retains its 0 value in the first $\hat{\imath} - 1$ iterations of the `for`-loop in Step 3, since the value of the `counter` field of $\mathcal{C}_i$ will not be equal to the number of its edges for any of $\mathcal{C}_1$, $\mathcal{C}_2, \ldots, \mathcal{C}_{\hat{\imath}-1}$ (Lemma 2.9). Then, in the $\hat{\imath}$th iteration (during which the component $\mathcal{C}_{\hat{\imath}}$ is processed), the `mark` field of $\mathcal{C}_i$ is set to 1 and $\mathcal{C}_i$ is oriented so that one of its edges points towards its endpoint which belongs to $V(\mathcal{C}_{\hat{\imath}})$. According to Lemma 4.1 (statement (i)), the latter implies that all the edges of $\mathcal{C}_i$ are oriented towards their endpoint which belongs to $V(\mathcal{C}_{\hat{\imath}})$. This orientation will not change in subsequent iterations of the `for`-loop of Step 3, since the `mark` field of $\mathcal{C}_i$ has been set to 1; nor will it change in Step 4. □

Then we can show the following result.

LEMMA 4.5. *When applied to a P_4-comparability graph, our orientation algorithm produces an acyclic P_4-transitive orientation.*

PROOF. The application of the recognition algorithm in Step 1 of the orientation algorithm and the fact that thereafter the inversion of the orientation of an edge causes the inversion of the orientation of all the edges in the same P_4-component imply that the resulting orientation is P_4-transitive. The proof will be complete if we show that it is also acyclic. Since the edges of the trivial P_4-components do not introduce cycles given that they are oriented according to a topological sorting of the vertices of the graph, it suffices to show that at the completion of Step 3 the directed subgraph spanned by the edges of the non-trivial P_4-components of the input graph G is acyclic. This follows directly from Lemmata 4.3 and 4.4. □

Time and Space Complexity. As described in the previous section, Step 1 of the algorithm can be completed in $O(nm)$ time. Step 2 takes $O(m)$ time by using bucket sorting, since there are $O(m)$ non-trivial P_4-components. Since the degree of a vertex of the graph does not exceed $n-1$, the total number of edges processed while processing the P_4-component $\mathcal{C}_i$ in Step 3 is $O(n\,|V(\mathcal{C}_i)|)$, which is $O(n\,(|E(\mathcal{C}_i)|+1)) = O(n\,|E(\mathcal{C}_i)|)$,

because the component $\mathcal{C}_i$ is connected (Lemma 2.1, statement (ii)) and hence $|E(\mathcal{C}_i)| \geq |V(\mathcal{C}_i)| - 1$. The time to process each such edge is $O(1)$, thus implying a total of $O(n\,|E(\mathcal{C}_i)|)$ time for the execution of Step 3 for the component $\mathcal{C}_i$; since an edge of the graph belongs to one P_4-component and a component is processed only once, the overall time for all the executions of Step 3 is $O(nm)$. Finally, Step 4 takes $O(n+m)$ time.

Summarizing, the time complexity of the orientation algorithm is $O(nm)$. The space complexity is linear in the size of the input graph.

From the above discussion, we obtain the following theorem.

THEOREM 4.1. *Let G be a P_4-comparability graph on n vertices and m edges. Then an acyclic P_4-transitive orientation of G can be computed in $O(nm)$ time and $O(n+m)$ space.*

Note that the input to our orientation algorithm does not need to be a P_4-comparability graph. If it is not, this will be detected in Step 1, and the algorithm will stop and will report it; otherwise, it will proceed, eventually computing the desired acyclic P_4-transitive orientation.

5. Concluding Remarks. In this paper we presented an $O(nm)$-time and linear-space algorithm for recognizing whether a graph of n vertices and m edges is a P_4-comparability graph, and an algorithm for computing an acyclic P_4-transitive orientation of a P_4-comparability graph which also requires $O(nm)$ time and linear space. Both algorithms exhibit the currently best time and space complexities to the best of our knowledge, and are simple enough to be easily used in practice. We also described a simple algorithm for computing the levels of the BFS-tree of the complement $\overline{G}$ of a graph G in time and space linear in the size of G, which we used in our P_4-comparability graph recognition algorithm.

The obvious open question is whether the P_4-comparability graphs can be recognized and/or oriented in $o(nm)$ time. It is worth mentioning that the P_4-indifference graphs can be recognized in linear time [11]. On the other hand, the currently best recognition algorithms for P_4-simplicial and Raspail graphs require $O(n^5)$ and $O(n^4)$ time, respectively [13]; a tighter analysis results in an $O(m^2)$-time algorithm in either case. So, it would be interesting to obtain $O(nm)$-time recognition algorithms for these two classes of graphs.

Moreover, it is worth investigating whether taking advantage of properties of the complement of the input graph can help establish improved algorithmic solutions for other problems as well; note that breadth-first and depth-first search on the complement of a graph can be executed in time linear in the size of the graph.

Acknowledgement. The authors would like to thank an anonymous referee whose suggestions helped improve the presentation of the paper.

Appendix

LEMMA. *Let $\mathcal{B}$ and $\mathcal{C}$ be two non-trivial P_4-components of a graph G such that $\mathcal{B}$ is of type B with respect to $\mathcal{C}$. If the subgraph of G spanned by the ribs of the P_4s in $\mathcal{C}$ is not bipartite, then $V(\mathcal{C}) \subset V(\mathcal{B})$.*

PROOF. First, we show that there exists a P_4 of $\mathcal{C}$ whose midpoints both belong to $V(\mathcal{B})$. Since the subgraph spanned by the ribs of the P_4s in $\mathcal{C}$ is not bipartite, there exists an odd cycle; let it be $v_1v_2\cdots v_{2k+1}$, for $k \geq 1$. Each edge of the cycle is the rib of a P_4 in $\mathcal{C}$. In particular, v_1v_2 is a rib. Then Lemma 2.10 (statement (iv)) implies that at least one of v_1, v_2 belongs to $V(\mathcal{B})$; suppose without loss of generality that $v_1 \in V(\mathcal{B})$. If $v_2 \in V(\mathcal{B})$, then any P_4 with v_1v_2 as its rib has both midpoints in $V(\mathcal{B})$, as desired. If $v_2 \notin V(\mathcal{B})$, then $v_3 \in V(\mathcal{B})$ since at least one of the vertices of the rib v_2v_3 has to belong to $V(\mathcal{B})$. Repeating this argument over and over along the cycle, we either find a rib v_iv_{i+1} such that $v_i, v_{i+1} \in V(\mathcal{B})$, or we have that $v_{2k+1} \in V(\mathcal{B})$, in which case the rib $v_{2k+1}v_1$ is such that $v_1, v_{2k+1} \in V(\mathcal{B})$. Therefore, there exists a P_4 of $\mathcal{C}$ whose midpoints both belong to $V(\mathcal{B})$; let this P_4 be $abcd$. In fact, a and d belong to $V(\mathcal{B})$ as well, as implied by Lemma 2.10 (statement (iii)) and by the fact that $c \in V(\mathcal{B})$ and $b \in V(\mathcal{B})$, respectively.

Next, we show that any P_4 ρ of $\mathcal{C}$ has all its vertices in $V(\mathcal{B})$; we apply induction on the length of the sequence of adjacent P_4s from $abcd$ to ρ. In the basis step we have a sequence of length 1; this is a sequence from the P_4 $abcd$ to $abcd$, and the proposition trivially holds. Suppose, for the induction hypothesis, that all the P_4s such that there exists a sequence of adjacent P_4s from $abcd$ to them of length i have all their vertices in $V(\mathcal{B})$. We will show that the same holds for any P_4 $a'b'c'd'$ such that there exists a sequence of adjacent P_4s from $abcd$ to $a'b'c'd'$ of length $i+1$. Let

$$abcd,\ a_2b_2c_2d_2,\ \ldots,\ a_ib_ic_id_i,\ a'b'c'd'$$

be the corresponding sequence. By the inductive hypothesis, $a_i, b_i, c_i, d_i \in V(\mathcal{B})$. Because $\mathcal{C}$ is separable (Lemma 2.10, statement (i)), the adjacency of the P_4s $a_ib_ic_id_i$ and $a'b'c'd'$ implies that $a' = a_i$ and $b' = b_i$, or $b' = b_i$ and $c' = c_i$, or $c' = c_i$ and $d' = d_i$. We examine each case separately.

Case (*a*): $a' = a_i$ *and* $b' = b_i$. Trivially, $a', b' \in V(\mathcal{B})$. Moreover, Lemma 2.10 (statement (iii)) implies that $c' \in V(\mathcal{B})$ (since $a' \in V(\mathcal{B})$) and that $d' \in V(\mathcal{B})$ (since $b' \in V(\mathcal{B})$).

Case (*b*): $b' = b_i$ *and* $c' = c_i$. Trivially, $b', c' \in V(\mathcal{B})$. Moreover, $c' \in V(\mathcal{B})$ implies that $a' \in V(\mathcal{B})$, and $b' \in V(\mathcal{B})$ implies that $d' \in V(\mathcal{B})$.

Case (*c*): $c' = c_i$ *and* $d' = d_i$. The case is similar to case (a).

Therefore, all vertices of the P_4 $a'b'c'd'$ belong to $V(\mathcal{B})$. Thus, the vertices of all the P_4s of $\mathcal{C}$ such that there exists a sequence of adjacent P_4s from $abcd$ to them have all four of their vertices in $V(\mathcal{B})$. Since every P_4 in $\mathcal{C}$ is such (Lemma 2.1, statement (i)), then $V(\mathcal{C}) \subseteq V(\mathcal{B})$. To show that $V(\mathcal{C}) \subset V(\mathcal{B})$, one need only observe that any P_4 of type (6) with respect to $\mathcal{C}$ contains two vertices of the graph G which do not belong to $V(\mathcal{C})$. □

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[2] S.R. Arikati and U.N. Peled, A polynomial algorithm for the parity path problem on perfectly orderable graphs, *Discrete Appl. Math.* **65**, 5–20, 1996.

[3] A. Brandstädt, V.B. Le, and J.P. Spinrad, *Graph Classes*: *A Survey*, Monographs on Discrete Mathematics and Applications 3, SIAM, Philadelphia, PA, 1999.
[4] V. Chvátal, Perfectly ordered graphs, *Annals Discrete Math*. **21**, 63–65, 1984.
[5] E. Dahlhaus, J. Gustedt and R.M. McConnell, Efficient and practical algorithms for sequential modular decomposition, *J. Algorithms* **41**, 360–387, 2001.
[6] E. Dahlhaus, J. Gustedt and R.M. McConnell, Partially complemented representations of digraphs, *Discrete Math. Theoret. Comput. Sci*. **5**, 147–168, 2002.
[7] C.M.H. de Figueiredo, J. Gimbel, C.P. Mello, and J.L. Szwarcfiter, Even and odd pairs in comparability and in P_4-comparability graphs, *Discrete Appl. Math*. **91**, 293–297, 1999.
[8] P.C. Gilmore and A.J. Hoffman, A characterization of comparability graphs and of interval graphs, *Canad. J. Math*. **16**, 539–548, 1964.
[9] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.
[10] M.C. Golumbic, D. Rotem, and J. Urrutia, Comparability graphs and intersection graphs, *Discrete Math*. **43**, 37–46, 1983.
[11] M. Habib, C. Paul, and L. Viennot, Linear-time recognition of P_4-indifference graphs, *Discrete Math. Theoret. Comput. Sci*. **4**, 173–178, 2001.
[12] C.T. Hoàng, Efficient algorithms for minimum weighted colouring of some classes of perfect graphs, *Discrete Appl. Math*. **55**, 133–143, 1994.
[13] C.T. Hoàng and B.A. Reed, Some classes of perfectly orderable graphs, *J. Graph Theory* **13**, 445–463, 1989.
[14] C.T. Hoàng and B.A. Reed, P_4-comparability graphs, *Discrete Math*. **74**, 173–200, 1989.
[15] H. Ito and M. Yokoyama, Linear time algorithms for graph search and connectivity determination on complement graphs, *Inform. Process. Lett*. **66**, 209–213, 1998.
[16] R.M. McConnell and J. Spinrad, Linear-time transitive orientation, *Proc*. 8*th ACM–SIAM Symp. on Discrete Algorithms* (*SODA* '97), pp. 19–25, 1997.
[17] M. Middendorf and F. Pfeiffer, On the complexity of recognizing perfectly orderable graphs, *Discrete Math*. **80**, 327–333, 1990.
[18] S.D. Nikolopoulos and L. Palios, Recognition and orientation algorithms for P_4-comparability graphs, *Proc*. 12*th Symp. on Algorithms and Computation* (*ISAAC* '01), pp. 320–331, 2001.
[19] T. Raschle and K. Simon, On the P_4-components of graphs, *Discrete Appl. Math*. **100**, 215–235, 2000.