RECOGNIZING BIPOLARIZABLE AND P_4-SIMPLICIAL GRAPHS

S.D. Nikolopoulos and L. Palios

07– 2003

Preprint, no 07 – 07 / 2003

Department of Computer Science
University of Ioannina
45110 Ioannina, Greece

Recognizing Bipolarizable and P_4-simplicial Graphs

Stavros D. Nikolopoulos and Leonidas Palios

Department of Computer Science, University of Ioannina
P.O.Box 1186, GR-45110 Ioannina, Greece
{stavros, palios}@cs.uoi.gr

Abstract: The classes of Raspail (also known as Bipolarizable) and P_4-simplicial graphs were introduced by Hoàng and Reed who showed that both classes are perfectly orderable and admit polynomial-time recognition algorithms [16]. In this paper, we consider the recognition problem on these classes of graphs and present algorithms that solve it in $O(nm)$ time. In particular, we prove properties and show that we can produce bipolarizable and P_4-simplicial orderings on the vertices of the input graph G, if such orderings exist, working only on P_3s that participate in a P_4 of G. The proposed recognition algorithms are simple, use simple data structures and both require $O(n+m)$ space. Additionally, we show how our recognition algorithms can be augmented to provide certificates, whenever they decide that G is not bipolarizable or P_4-simplicial; the augmentation takes $O(n+m)$ time and space. Finally, we include a diagram on class inclusions and the currently best recognition time complexities for a number of perfectly orderable classes of graphs and some preliminary results on forbidden subgraphs for the class of P_4-simplicial graphs.

Keywords: Bipolarizable (Raspail) graph, P_4-simplicial graph, perfectly orderable graph, recognition, algorithm, complexity, forbidden subgraph.

1 Introduction

A linear order $\prec$ on the vertices of a graph G is *perfect* if the ordered graph $(G, \prec)$ contains no induced P_4 $abcd$ with $a \prec b$ and $d \prec c$ (such a P_4 is called an *obstruction*). In the early 1980s, Chvátal [4] defined the class of graphs that admit a perfect order and called them *perfectly orderable* graphs.

Chvátal proved that if a graph G admits a perfect order $\prec$, then the greedy coloring algorithm applied to $(G, \prec)$ produces an optimal coloring using only $\omega(G)$ colors, where $\omega(G)$ is the clique number of G. This implies that the perfectly orderable graphs are perfect; a graph G is *perfect* if for each induced subgraph H of G, the chromatic number $\chi(H)$ equals the clique number $\omega(H)$ of the subgraph H. The class of perfect graphs was introduced in the early 1960s by Berge [1], who also conjectured that a graph is perfect if and only if it contains no induced subgraph isomorphic to an odd cycle of length at least five, or to the complement of such an odd cycle. This conjecture, known as the *strong perfect graph conjecture*, has been recently established due to the work of Chudnovsky *et al.* [3].

It is well-known that many interesting problems in graph theory, which are NP-complete in general graphs, have polynomial-time solutions in graphs that admit a perfect order [2, 8]; unfortunately, it is NP-complete to decide whether a graph admits a perfect order [22]. Since the recognition of

Our main objective is to study the recognition problem on the classes of bipolarizable and P_4-simplicial graphs and we present $O(nm)$-time algorithms for each of these problems. Our algorithms rely on properties that we establish and which allow us to only work with P_3s of the input graph G which participate in P_4s of G; such P_3s can be computed in $O(nm)$ time by means of the BFS-trees of the *complement* of the graph rooted at each of its vertices [23]. The proposed recognition algorithms are simple, use simple data structures and both require $O(n+m)$ space. Additionally, we describe how to augment our two recognition algorithms so that they return a certificate whenever they decide that G is not bipolarizable or P_4-simplicial, thus, providing the most natural evidence that the input graph G indeed is not bipolarizable or P_4-simplicial. In particular, for the case of bipolarizable graphs, the augmented algorithm returns a forbidden subgraph contained in G. The augmented algorithms take $O(n+m)$ additional time and $O(n+m)$ space. Finally, we give class inclusion results for a number of perfectly orderable classes of graphs and show the currently best time complexities to recognize members of these classes and also present results on forbidden subgraphs for the class of P_4-simplicial graphs.

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. The recognition algorithms for bipolarizable and P_4-simplicial graphs are described and analyzed in Sections 3 and 4, respectively. Sections 5 and 6 give results on class inclusions for a number of perfectly orderable classes and on forbidden subgraphs for the class of P_4-simplicial graphs, and Section 7 concludes with a summary of our results and some open problems.

2 Theoretical Framework

We consider finite undirected graphs with no loops or multiple edges. Let G be such a graph; then, $V(G)$ and $E(G)$ denote the set of vertices and of edges of G respectively. The *neighborhood* $N(x)$ of a vertex $x \in V(G)$ is the set of all the vertices of G which are adjacent to x. The *closed neighborhood* of x is defined as $N[x] := \{x\} \cup N(x)$. The subgraph of G induced by a subset S of G's vertices is denoted by $G[S]$. A subset $A \subseteq V(G)$ of p vertices is a *p-clique*, or *clique*, if it induces a complete subgraph, i.e., $G[A] = K_p$; a single vertex is a 1-clique. An *independent set* is a subset $B \subseteq V(G)$ of vertices no two of which are adjacent; it is also called *stable set*. A subset $H \subseteq V(G)$ of vertices is *homogeneous* if $2 \leq |H| < |V(G)|$ and each vertex $x \in V(G) - H$ sees either all vertices or no vertex in H, i.e., either $H \subseteq N(x)$ or $H \cap N(x) = \emptyset$.

A *path* in a graph G is a sequence of vertices $v_0 v_1 \ldots v_k$ 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 may be *undirected* or *directed* depending on whether G is an undirected or 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) $v_0 v_1 \ldots v_k$ is called a *cycle* (*simple cycle*) of length $k+1$ if $v_0 v_k \in E(G)$. A simple path (cycle) $v_0 v_1 \ldots v_k$ is *chordless* if $v_i v_j \notin E(G)$ for any two non-consecutive vertices v_i, v_j in the path (cycle). The chordless path (chordless cycle, respectively) on n vertices is commonly denoted by P_n (C_n, respectively). In particular, a chordless path on 4 vertices is denoted by P_4.

Let *abcd* be a P_4 of a graph. The vertices b and c are called *midpoints* and the vertices a and d *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 the P_4 *abcd*, the edge *bc* is its rib and the edges *ab* and *cd* are its wings.

Our bipolarizable graph recognition algorithm relies on the result stated in the following lemma.

Lemma 2.1. *Let G be a graph that contains no induced subgraph isomorphic to a house graph ($\overline{P}_5$) or the graphs A and D_6 of Figure 1. Then, G does not contain a C_4 abcd such that abc and bcd are P_3s participating in P_4s of G.*

Proof: Suppose for contradiction that G contains a C_4 *abcd* meeting the conditions in the statement of the lemma. We distinguish cases. Suppose first that the P_3 *abc* participates in the P_4 *abcx* and that

Lemma 3.1. *Let G be a bipolarizable graph and let abc be a P_3 participating in a P_4 of G. If bcd is another such P_3, then G contains the P_4 $abcd$.*

Proof: If the path $abcd$ is not a P_4 then G must contain the edge ad. But this creates a C_4 meeting the conditions of Lemma 2.1; a contradiction. ∎

Then, Lemma 3.1 implies the following corollary.

Corollary 3.1. *Let G be a bipolarizable graph and let F be the orientation of G that results from the bipolarizable ordering of the vertices of G (i.e., the wings of each P_4 are oriented towards the P_4's endpoints). Then, for each edge bc of G for which there exist P_3s abc and bcd participating in P_4s of G, the edges ab and cd (for all such a and d) get oriented towards a and d respectively.*

Proof: Let us consider any such P_3 abc; then, because of the existence of the P_3 bcd, Lemma 3.1 applies, and thus $abcd$ is a P_4 of G. Therefore, the edge ab is oriented towards a in F, and this holds for all such a, and the edge cd is oriented towards d in F and this holds for all such d. ∎

The algorithm for the recognition of bipolarizable graphs applies Corollary 3.1. The input graph G is assumed to be given in adjacency list representation. The algorithm uses two arrays, an array $M[\,]$ and an array $S[\,]$, of size $2m$ each. The array $M[\,]$ has entries $M[xy]$ and $M[yx]$, for each edge xy of G; the entry $M[xy]$ is equal to 1 if there exist P_3s xyz participating in P_4s of G, and is equal to 0 otherwise. As a result, for an edge xy, both $M[xy]$ and $M[yx]$ are equal to 1 iff there exist P_3s xyz and txy participating in P_4s of G. The array $S[\,]$ too has entries $S[xy]$ and $S[yx]$, for each edge xy of G; the entry $S[xy]$ is equal to the index number of the partition set of $N(y)$ to which x belongs (see Lemma 2.3). As a result, a path xyz is a P_3 participating in P_4s of G iff $S[xy] \neq 0$, $S[zy] \neq 0$, and $S[xy] \neq S[zy]$. In more detail, the algorithm works as follows.

Bipolarizable Graph Recognition Algorithm

Input: an undirected graph G on n vertices and m edges.

Output: yes, if G is a bipolarizable graph; otherwise, no.

1. Initialize the entries of the arrays $M[\,]$ and $S[\,]$ to 0; for each vertex v, sort the records of the neighbors of v in v's adjacency list in increasing vertex index number;

2. Find all the P_3s participating in P_4s of G; for each such P_3 abc, set the entries $M[ab]$ and $M[cb]$ equal to 1, and update appropriately the entries $S[ab]$ and $S[cb]$;

3. For each edge uv of G such that $M[uv] = 1$ and $M[vu] = 1$ do

   3.1 traverse the adjacency lists of u and v in lockstep fashion and process the neighbors of v which are not neighbors of u, and the neighbors of u which are not neighbors of v as follows:

   3.2 for each neighbor w of v which is not adjacent to u do
   if $S[uv] \neq 0$ and $S[wv] \neq 0$ and $S[uv] \neq S[wv]$
   then {*uvw is a P_3 in a P_4 of G*}
   if the edge vw has not received an orientation
   then orient it towards w;
   else if it is oriented towards v
   then print that G is not a bipolarizable graph; exit.

   3.3 for each neighbor w of u which is not adjacent to v do
   if $S[vu] \neq 0$ and $S[wu] \neq 0$ and $S[vu] \neq S[wu]$
   then {*vuw is a P_3 in a P_4 of G*}
   if the edge uw has not received an orientation
   then orient it towards w;
   else if it is oriented towards u
   then print that G is not a bipolarizable graph; exit.

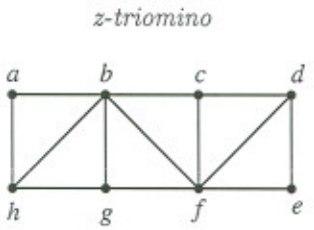

Figure 2

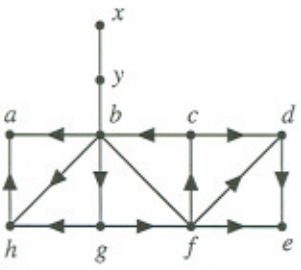

Figure 3

A *k-wheel* ($k \geq 3$) is the graph formed by a set of $3k$ vertices, namely, $v_0, v_1, \ldots, v_{k-1}$, $r_0, r_1, \ldots, r_{k-1}$, and $s_0, s_1, \ldots, s_{k-1}$, such that

- ▷ the vertices $v_0, v_1, \ldots, v_{k-1}$ form a clique, while each of the sets $\{r_0, \ldots, r_{k-1}\}$ and $\{s_0, \ldots, s_{k-1}\}$ is an independent set,
- ▷ for $0 \leq i, j \leq k-1$, v_i is adjacent to r_j except for $j = i+1$,
- ▷ for $0 \leq i, j \leq k-1$, v_i is adjacent to s_j except for $j = i, i+1$, and
- ▷ for $0 \leq i \leq k-1$, r_i is adjacent to s_i but non-adjacent to any s_j for $j \neq i$,

where all integer subscripts are taken modulo k.

(We note that the 2-wheel is also well defined and coincides with P_6.)

As indicated by the algorithm, there are two reasons due to which a given graph G can be found non-bipolarizable. First, *a conflict of orientation* may arise on an edge of G which ends up receiving opposite orientations by different P_4s sharing it. Second, if no conflict has arisen, there may be the case that the final orientation assignment contains a *directed cycle*. Relating these two cases to the forbidden subgraphs, it is not difficult to see that:

Lemma 3.2. *Let G be a graph containing any of the six graphs of Figure 1. Then, G exhibits a conflict of orientation.*

Proof: Clearly true; the bottom horizontal edge of each of these graphs receives opposite orientations. ∎

We also note that a conflict of orientation may arise if two z-triominoes, a z-triomino and a k-wheel, or two k-wheels share an edge and impose on it opposite orientations. In such a case, however, it is not difficult to see that the P_4s that forced these orientations would form one of the graphs of Figure 1.

Lemma 3.3. *Let G be a graph which contains a z-triomino as shown in Figure 2 but none of the six graphs of Figure 1 or a k-wheel. Then, G exhibits a directed C_4 $bcfg$ with a single undirected diagonal bf.*

Proof: Since G does not contain any of the graphs in Figure 1, each edge of G will receive at most one orientation. Thus, the edges of each z-triomino of G receive the orientations indicated in Figure 3; that is, for the z-triomino with vertices a, b, c, d, e, f, g, h, a single directed cycle $bcfg$ is formed. It suffices to show that the diagonal bf is not the wing of any P_4 of G and thus receives no orientation. Clearly, it does not get oriented due to a P_4 induced by a subset of the vertices of the z-triomino. Considering P_4s with vertices in addition to the vertices of the triomino is exhausted, due to symmetry, to the following three cases:

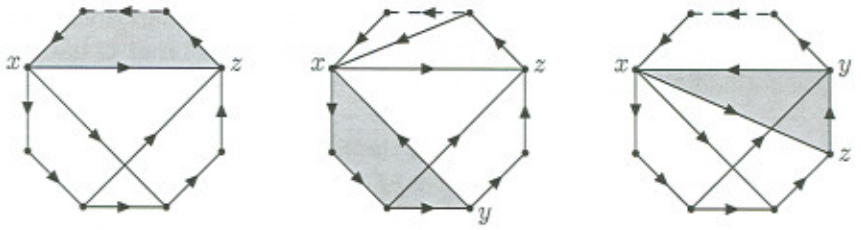

Figure 4

if no such vertex y exists
    find the vertex z, if any, such that the edge xz is directed towards z, and z exhibits the largest "rank" larger than x's;
    if such a vertex z exists
        clip the list L by removing any vertex records between x and z;
else
    find the vertex z, if any, such that the edge xz is directed towards z, and z exhibits the largest "rank" larger than x's and smaller than y's;
    if no such vertex z exists
        clip the list L by removing any vertex records to the left of x and to the right of y;
    else
        clip the list L by maintaining the vertex record for x followed by those between z and y inclusive;

The vertices that remain in the list L after the procedure `get_c-d-cycle(`L`)` has completed induce a directed cycle without directed chords in the directed subgraph $\overrightarrow{G}$. The correctness of the computation follows easily: the three main cases are illustrated in Figure 4, where the dashed edge indicates the edge directed from the last vertex of the list L to its first vertex and the shaded region is the clipped list L after the processing of vertex x. It is important to observe that the clipping indeed maintains the ordering of the "ranks" of the vertices along the list L, and that after a vertex x has been processed the resulting list L induces a directed cycle without directed chords incident on x.

Locating a z-triomino. In case the vertices in the resulting list L returned by the procedure `get_c-d-cycle()` form a C_4 $bcfg$ that has a single diagonal bf, we can obtain the z-triomino built around the C_4 (see Figure 2) as follows:

1. store the neighbors of the vertices b, c, f, g in an array each for constant time adjacency tests;
2. for each vertex d adjacent to c do
    if $d \neq b$ and $d \notin N(b)$ and $d \in N(f)$ and $d \notin N(g)$
        for each vertex e adjacent to d do
            if $e \notin N(b)$ and $e \notin N(c)$ and $e \in N(f)$ and $e \notin N(g)$
                the vertices d and e have been located;
3. work similarly as in Step 2 in order to locate the vertices a and h;
4. the vertices a, b, c, d, e, f, g, h induce a z-triomino in G.

We note that we do not need to check whether $d \neq f$ and $e \neq f$ as these are precluded from the fact that $d, e \notin N(b)$. To see that the subgraph induced by the vertices $a, b, \ldots, h$ is a z-triomino in G, note that by construction the vertices a, d, e, h have the desired adjacency relations to the vertices b, c, f, g. The pairwise adjacencies of a, d, e, h need not be checked; the fact that these four vertices are all distinct implies that if the pairwise adjacencies were not as indicated in the z-triomino, then

Finally, regarding the procedure to locate a k-wheel, we have that Steps 1, 3 and 5 take $O(n)$ time, Step 2 takes $O(n+m)$ time (the number p_x is computed by traversing the adjacency list of x and counting the number of x's neighbors in the set P), while Step 4 takes $O(\sum_i \sum_w deg(w)) = O(\sum_{x \in V(G)} deg(w)) = O(m)$ time since the sets R_i are disjoint.

The total space required is linear in the size of the input graph. Therefore, we have the following result:

Theorem 3.2. *Let G be an undirected graph on n vertices and m edges. The bipolarizable graph recognition algorithm presented in this section can be augmented to provide a forbidden subgraph in G, whenever it decides that G is not bipolarizable, in $O(n+m)$ time and $O(n+m)$ space.*

4 Recognition of P_4-simplicial Graphs

Our P_4-simplicial graph recognition algorithm relies on the corresponding algorithm of Hoàng and Reed [16]; our contribution is that we restate the main condition on which their algorithm is based in terms of P_3s participating in P_4s of the input graph, and we show how to efficiently take advantage of it in order to achieve an $O(nm)$-time complexity. As described in the introduction, their algorithm works as follows: it initially sets $H := V(G)$ and then it iteratively identifies a vertex x in H such that G does not contain a P_4 of the form $abxc$ with $b, c \in H$, and removes it from H; the graph G is P_4-simplicial iff the above process continues until H becomes the empty set.

It is not difficult to see that the property a vertex x has to have in order to be removed from H can be equivalently stated as follows:

Property 4.1. *Let H be the current set of vertices of a given graph G. Then, a vertex x can be removed from H if and only if there does not exist any P_3 bxc participating in a P_4 of G with $b, c \in H$.*

In light of Property 4.1, we can obtain an algorithm for deciding whether a given graph G is P_4-simplicial by keeping count, for each vertex $v \in H$, of the number of P_3s bvc with $b, c \in H$ which participate in P_4s of G, and by removing a vertex x from H whenever the number of such P_3s associated with x is 0. The proposed algorithm implements precisely this strategy; it takes advantage of the computation of the P_3s in P_4s of G in $O(nm)$ time, and maintains an array $NumP3[\,]$ of size n, which stores for each vertex v in H the number of P_3s bvc which participate in P_4s of G and have $b, c \in H$. The input graph G is assumed to be given in adjacency list representation. In more detail, the algorithm works as follows.

P_4-simplicial Graph Recognition Algorithm

Input: an undirected graph G on n vertices and m edges.

Output: yes, if G is a P_4-simplicial graph; otherwise, no.

1. Collect all the vertices of G into a set H;
   make a copy $A[v]$ of the adjacency list of each vertex v of G while attaching at each record of the list an additional field *set*;
2. For each vertex v of G do
   - 2.1 compute the partition of the vertices in $N(v)$ into sets $S_0, S_1, \ldots, S_{k_v}, S_{k_v+1}$ as described in Lemma 2.3, and update appropriately the fields *set* of the records in the adjacency list $A[v]$ of v;
   - 2.2 compute the number of P_3s avb participating in P_4s of G and assign this number to $NumP3[v]$;

Step 5 $O(n)$ time. Step 2 takes $O(nm)$ time [23], while Step 3 takes $O(n)$ time. As a vertex is inserted at most once in the list L, the time complexity of Step 4 is $O\left(\sum_x \left(1 + \sum_{u \in N(x)} deg(u)\right)\right)$, where $deg(u)$ denotes the degree of u in G. Since $\sum_{u \in N(x)} deg(u) = O(m)$, the time complexity of Step 4 is $O(nm)$. The computation of the P_3s participating in P_4s takes linear space, and thus the total space needed by the recognition algorithm is clearly linear in the size of the input graph G.

Summarizing, we obtain the following theorem.

Theorem 4.1. *Let G be an undirected graph on n vertices and m edges. Then, it can be determined whether G is a P_4-simplicial graph in $O(nm)$ time and $O(n+m)$ space.*

4.1 Providing a Certificate

As in the case of the bipolarizable graph recognition algorithm, the above algorithm can be made to return a certificate whenever it decides that the input graph G is not P_4-simplicial. In particular, it could return the at the time (non-empty) value of the set H, which would indicate a subgraph of G none of whose vertices can be removed in the sense of Property 4.1. Clearly, this does not require any additional computation time and space.

However, it would be more interesting if the algorithm located a minimal such subset H' of H, that is, a subset H' such that every vertex y of H' forms a P_3 xyz participating in a P_4 of the input graph with $x, z \in H'$. To see the benefits of this, consider for example that the input graph G contained two domino graphs sharing an edge. Since the domino graph is a forbidden subgraph for the class of P_4-simplicial graphs, the algorithm would stop, would report that G is not P_4-simplicial and would return a set H of vertices which would be a superset of the set of vertices of both domino graphs. If however a minimal such set of vertices were returned, then one would be very close to identifying a forbidden subgraph in G. This approach, although very interesting, is hindered in part by the fact that no complete characterization of the P_4-simplicial graphs by forbidden subgraphs is currently available in the literature. A first attempt on such a characterization can be found in Section 6.

5 Class Inclusions and Recognition Time Complexities

Figure 5 shows a diagram of class inclusions for a number of perfectly orderable classes of graphs and the currently best time complexities to recognize members of these classes. For definitions of the classes shown, see [2, 8]; note that the P_4-free and the chordal graphs are also known as co-graphs and triangulated graphs respectively. In the diagram, there exists an arc from a class $\mathcal{A}$ to a class $\mathcal{B}$ if and only if $\mathcal{B}$ is a proper subset of $\mathcal{A}$. Hence, if any two classes are not connected by an arc, then each of these classes contains graphs not belonging to the other class (there are such sample graphs for each pair of non-linked classes).

Most of these class inclusions can be found in [2] where a similar diagram with many more graph classes appears; Figure 5 comes from a portion of the diagram in [2] augmented with the introduction of the inclusion relations for the classes of P_4-simplicial, bipolarizable, and P_4-indifference graphs, as described in Lemmas 5.1-5.3. We will show next that the class of weak bipolarizable graphs [24] is a proper subset of the class of P_4-simplicial graphs. In fact, we show a slightly stronger result as established in the following proposition.

Proposition 5.1. *Let G be a weak bipolarizable graph and let v be a vertex of G. Then, G admits a P_4-simplicial order $\prec$ on its vertices such that $v \prec x$ for any vertex x of G other than v.*

Proof: We apply induction on the size of the graph by taking advantage of Theorem 1 of [24] which states that a graph G is weak bipolarizable if and only if every induced subgraph of G is chordal or contains a homogeneous set. For the basis step, it is not difficult to verify that every weak bipolarizable graph on up to 3 vertices admits a P_4-simplicial order as described, since G does not contain any P_4s.

Regarding the relation of P_4-simplicial and the HHD-free and co-chordal graphs, note that the graph D_6 of Figure 1 is both HHD-free and co-chordal but is not P_4-simplicial whereas the house graph and P_5 are P_4-simplicial but not HHD-free and not co-chordal respectively.

Lemma 5.2. *The class of bipolarizable graphs is a proper subset of the class of weak bipolarizable graphs and a proper superset of the classes of P_4-sparse and split graphs.*

Proof: The fact that *Bipolarizable* $\subset$ *Weak Bipolarizable* has been established in [24]. To establish the relationship of P_4-sparse and bipolarizable graphs, we note that none of the forbidden subgraphs for the class of bipolarizable graphs is P_4-sparse; see Figures 1 and 2 and note that a k-wheel of order $k \geq 2$ contains the P_5 $s_0 r_0 v_0 v_{k-1} r_1$. This implies that any graph which is not bipolarizable cannot be P_4-sparse, or conversely that *P_4-sparse* $\subseteq$ *Bipolarizable*. The proper inclusion follows from the fact that a P_5 is bipolarizable but not P_4-sparse.

It is not difficult to see that *Split* $\subseteq$ *Bipolarizable*; the vertex set of a split graph can be partitioned into an independent set and a clique, which implies that any P_4 of a split graph has its midpoints in the clique and its endpoints in the independent set. Thus any ordering of the vertices of a split graph where all the vertices of the clique precede all the vertices of the independent set gives a bipolarizable ordering of the graph. The proper inclusion follows from the fact that C_4 is bipolarizable but not split. ∎

Lemma 5.3. *The class of P_4-indifference graphs is a proper subset of the class of weak bipolarizable graphs and a proper superset of the class of P_4-reducible graphs.*

Proof: The fact that *P_4-indifference* $\subset$ *Weak Bipolarizable* follows from the fact that the set of forbidden subgraphs for the class of weak bipolarizable graphs is a proper subset of the set of forbidden subgraphs for the class of P_4-indifference graphs (compare [24] and [15]).

To see that *P_4-reducible* $\subseteq$ *P_4-indifference*, we recall that every vertex of a P_4-reducible belongs to at most one P_4, which implies that the P_4s of a P_4-reducible graph are vertex-disjoint. Thus, we can create a linear order of the vertices of such a graph by concatenating the vertices of each P_4 at a time, in the order they appear along the P_4, and by appending any remaining vertices; then, the resulting ordering is a P_4-indifference ordering of the vertices of the graph. The proper inclusion follows from the fact that the P_5 is a P_4-indifference graph but not P_4-reducible. ∎

The non-inclusion relation between bipolarizable and co-chordal graphs follows from the counterexamples for the non-inclusion relation of the P_4-simplicial and co-chordal graphs. A non-inclusion relation also holds for the bipolarizable and the chordal graphs (consider a C_4 and the z-triomino) and for the bipolarizable and the P_4-indifference graphs (consider the forbidden subgraphs F_5 of [15] and the z-triomino).

Figure 5 has also partitioned the depicted classes of graphs based on the time complexities of the currently best recognition algorithms: see [7, 26] for the $O(\min\{n^3 \log^2 n, m^2\})$-time complexity range, [14, 18] for the $O(\min\{n^3, m^2\})$-time complexity range, [23, 24] for the $O(nm)$-time complexity range, and [10, 19, 20, 5, 25, 9, 11] for the $O(n+m)$-time range. We note that the algorithm of [14] for the recognition of HHD-free graphs has a stated time complexity of $O(n^4)$; this can be easily seen to be $O(m^2)$ if the number m of edges of the graph is taken into account. Similarly, the algorithm of [24] for the recognition of weak bipolarizable graphs has a stated time complexity of $O(n^3)$; since $O(n+m)$ time suffices to determine whether a graph is chordal and to compute a homogeneous set (by means of modular decomposition [21, 6]), if one exists, the stated time complexity can be seen to be $O(nm)$.

7 Concluding Remarks

We have presented recognition algorithms for the classes of bipolarizable (also known as Raspail) and P_4-simplicial graphs running in $O(nm)$ time, where n and m are the number of vertices and of edges of the input graph. Our proposed algorithms are simple, use simple data structures and require $O(n+m)$ space; the algorithms can also be augmented so that they return a certificate, whenever they decide that the input graph is not bipolarizable or P_4-simplicial, in $O(n+m)$ additional time and space. We have also presented results on class inclusions and recognition time complexities for a number of perfectly orderable classes of graphs, and also some results on forbidden subgraphs for the class of P_4-simplicial graphs.

We leave as an open problem the designing of $o(nm)$-time algorithms for recognizing bipolarizable and/or P_4-simplicial graphs. In light of the linear-time recognition of P_4-indifference graphs [10], it would be worth investigating whether the recognition of P_4-comparability, P_4-simplicial, and bipolarizable graphs is inherently more difficult; it must be noted that the approach used in [10] is different from those used for the recognition of the remaining classes as it reduces in part the problem to the recognition of interval graphs which can be carried out in linear time. Finally, another interesting open problem is that of obtaining a complete characterization of the P_4-simplicial graphs by forbidden subgraphs.

References

[1] C. Berge, Färbung von Graphen deren sämtliche bzw. deren ungerade Kreise starr (Zusammenfassung), *Wissenschaftliche Zeitschrift*, Martin Luther Universität Halle-Witterberg, Mathematisch-Naturwissenschaftliche Reihe, 114–115, 1961.

[2] A. Brandstädt, V.B. Le, and J.P. Spinrad, *Graph classes: A survey*, SIAM Monographs on Discrete Mathematics and Applications, 1999.

[3] M. Chudnovsky, N. Robertson, P.D. Seymour, and R. Thomas, The strong perfect graph theorem, *submitted.*

[4] V. Chvátal, Perfectly ordered graphs, *Annals of Discrete Math.* **21**, 63–65, 1984.

[5] D.G. Corneil, Y. Perl, and L.K. Stewart, A linear recognition algorithm for cographs, *SIAM J. Comput.* **14**, 926–934, 1985.

[6] E. Dahlhaus, J. Gustedt, and R.M. McConnell, Efficient and practical algorithms for sequential modular decomposition, *J. Algorithms* **41**, 360–387, 2001.

[7] E.M. Eschen, J.L. Johnson, J.P. Spinrad, and R. Sritharan, Recognition of some perfectly orderable graph classes, *Discrete Appl. Math.* **128**, 355–373, 2003.

[8] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, Inc., 1980.

[9] M. Habib, R.M. McConnell, C. Paul, and L. Viennot, Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing, *Theoret. Comput. Sci.* **234**, 59–84, 2000.

[10] M. Habib, C. Paul, and L. Viennot, Linear time recognition of P_4-indifference graphs, *Discrete Math. and Theor. Comput. Sci.* **4**, 173–178, 2001.

[11] P.L. Hammer and B. Simeone, The splittance of a graph, *Combinatorica* **1**, 275–284, 1981.

[12] A. Hertz, Bipolarizable graphs, *Discrete Math.* **81**, 25–32, 1990.

[13] C.T. Hoàng, On the complexity of recognizing a class of perfectly orderable graphs, *Discrete Appl. Math.* **66**, 219–226, 1996.

[14] C.T. Hoàng and N. Khouzam, On brittle graphs, *J. Graph Theory* **12**, 391–404, 1988.

[15] C.T. Hoàng, F. Maffray, and M. Noy, A characterization of P_4-indifference graphs, *J. Graph Theory* **31**, 155–162, 1999.