Recognizing Bipolarizable and P_4-Simplicial Graphs*

Stavros D. Nikolopoulos and Leonidas Palios

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

Abstract. Hoàng and Reed defined the classes of Raspail (also known as Bipolarizable) and P_4-simplicial graphs, both of which are perfectly orderable, and proved that they 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, where n and m are the numbers of vertices and of edges of the input graph. In particular, we prove properties and show that we can produce bipolarizable and P_4-simplicial orderings on the vertices of a graph G, if such orderings exist, working only on P_3s that participate in P_4s of G. The proposed recognition algorithms are simple, use simple data structures and require $O(n + m)$ space. Moreover, we present 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 and studied by Berge [1], who also conjectured that a graph is perfect if and only if it has 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].

* Research partially funded by the European Commission and the Hellenic Ministry of Education through EPEAEK II.

H.L. Bodlaender (Ed.): WG 2003, LNCS 2880, pp. 358–369, 2003.

It is well-known that many interesting problems in graph theory, which are NP-complete in general graphs, have polynomial 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 perfectly orderable graphs is NP-complete, we are interested in characterizing graphs which form polynomially recognizable subclasses of perfectly orderable graphs. Many such classes of graphs, with very interesting structural and algorithmic properties, have been defined so far and shown to admit polynomial-time recognitions (see [2,8]); note however that not all subclasses of perfectly orderable graphs admit polynomial-time recognitions [13].

Hoàng and Reed [16] introduced four subclasses of perfectly orderable graphs, namely, the Raspail (also known as Bipolarizable), P_4-simplicial, P_4-indifference, and P_4-comparability graphs, and provided polynomial-time recognition algorithms for these four classes of graphs. A graph G is *bipolarizable* if it admits a linear order $\prec$ on its vertices such that every P_4 $abcd$ has either ($b \prec a$, $b \prec c$, $c \prec d$) or ($b \prec a$, $c \prec b$, $c \prec d$). A graph G is *P_4-simplicial* if it admits a linear order $\prec$ such that every P_4 has either a P_4-indifference ordering (i.e., every P_4 $abcd$ has either ($a \prec b$, $b \prec c$, $c \prec d$) or ($d \prec c$, $c \prec b$, $b \prec a$)) or a bipolarizable ordering. Note that every linear order $\prec$ on the vertices of a graph G yields an acyclic orientation of the edges, where each edge ab is oriented from a to b if and only if $a \prec b$. On the other hand, every acyclic orientation gives at least one linear order (for example, the order taken by a topological sorting). Hence, bipolarizable and P_4-simplicial graphs can also be defined in terms of orientations.

As mentioned in the previous paragraph, the recognition problem on both bipolarizable and P_4-simplicial graphs has been addressed by Hoàng and Reed [16]; for a graph on n vertices, their algorithms run in $O(n^4)$ and $O(n^5)$ time respectively. Recently, Eschen *et al.* [7] described recognition algorithms for several classes of perfectly orderable graphs, among which $O(n^{3.376})$-time algorithms for both bipolarizable and P_4-simplicial graphs. We note that Hoàng and Reed also presented algorithms which solve the recognition problem for P_4-indifference and P_4-comparability graphs which run in $O(n^6)$ and $O(n^4)$ time [16,17]; recent results on these problems include $O(n+m)$-time and $O(nm)$-time algorithms respectively [10,23], where m is the number of edges of the input graph.

In this paper, we consider the recognition problems for bipolarizable and P_4-simplicial graphs and present $O(nm)$-time algorithms for their solution. Our algorithms rely on properties that we establish and which allow us to work only with P_3s which participate in P_4s of the input graph G; such P_3s can be computed in $O(nm)$ time by means of the BFS-trees of the *complement* of G rooted at each of its vertices [23]. The proposed recognition algorithms are simple, use simple data structures and require $O(n+m)$ space. Furthermore, 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 finally we also present results on forbidden subgraphs for the class of P_4-simplicial graphs.

2 Preliminaries

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 subgraph of G induced by a subset S of G's vertices is denoted by $G[S]$. 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 $N[x] := \{x\} \cup N(x)$.

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 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.

Computing all the P_3s participating in P_4s of a graph G: In [23], it has been shown that all the P_3s participating in P_4s of a graph G on n vertices and m edges can be computed in $O(nm)$ time and $O(n+m)$ space as follows: for each vertex v, the BFS-tree $T_{\overline{G}}(v)$ of the *complement* of G rooted at v is constructed and the vertices in the 2nd level of the tree are partitioned into sets $S_1, \ldots, S_{k_v}$, where two vertices belong to the same S_i iff they have the same neighbors in the 1st level of $T_{\overline{G}}(v)$; the root v of $T_{\overline{G}}(v)$ is assumed to be located in the 0th level. Then, avb is a P_3 participating in a P_4 of G iff $ab \notin E(G)$ and either exactly one of a, b belongs to the 2nd level and the other to the 3rd level of $T_{\overline{G}}(v)$, or both a and b belong to the 2nd level but they are in different sets of the partition $S_1, \ldots, S_{k_v}$.

Since the vertices in the 2nd and 3rd level of $T_{\overline{G}}(v)$ form a subset of the neighborhood of v, we can give a more unified criterion for deciding whether a P_3 avb participates in a P_4 of G by defining the following partition of $N(v)$:

Definition 2.1. *For each vertex v of a graph G, we consider the following partition of the neighborhood $N(v)$ of v:*

- ▷ *the partition of the vertices in the 2nd level of $T_{\overline{G}}(v)$ into $S_1, \ldots, S_{k_v}$ as described above;*
- ▷ *all the vertices in the 3rd level of $T_{\overline{G}}(v)$ are placed in a set S_{k_v+1};*
- ▷ *all remaining vertices in $N(v)$ are placed in a set S_0 (no such vertex a forms a P_3 avb participating in P_4s of G for any vertex b of G).*

Then, for any $a, b \in N(v)$, avb is a P_3 participating in a P_4 of G iff $ab \notin E(G)$, and if $a \in S_i$ and $b \in S_j$ then $i \neq 0$, $j \neq 0$, and $i \neq j$.

Convention: Throughout the paper, we assume that the input graph G has n vertices and m edges and is given in adjacency list representation.

3 Recognition of Bipolarizable Graphs

The definition of bipolarizable graphs implies that they can be efficiently recognized as soon as the wings of all the P_4s have been computed. The method described in [23] for computing all the P_3s participating in P_4s of a given graph does not seem to extend to produce within the same time complexity which edge of the P_3 is the rib and which is the wing of the P_4. However, in the case of bipolarizable graphs, we establish a property that can be used for their efficient recognition. First, we need the following lemma:

Lemma 3.1. *Let G be a graph that contains no induced subgraph isomorphic to a house graph or the graphs F_1 and F_2 of Figure 2. Then, G contains 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 the P_3 *bcd* participates in the P_4 *bcdy*. Then, $xd \notin E(G)$, otherwise the vertices a, b, c, d, x would induce a house in G. In a similar fashion, $ya \notin E(G)$ either. But then, if $xy \notin E(G)$, then the subgraph induced by a, b, c, d, x, y is isomorphic to F_1 whereas if $xy \in E(G)$, it is isomorphic to F_2; a contradiction in either case. The remaining three cases (depending on whether *abc* participates in a P_4 *xabc* or *abcx* and on whether *bcd* participates in a P_4 *ybcd* or *bcdy*) are handled similarly. ∎

Since the bipolarizable graphs do not contain the house graph, F_1, or F_2 (and also some other subgraphs [12,16]), Lemma 3.1 implies the following corollary.

Corollary 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 3.1; a contradiction. ∎

(We note that Corollary 3.1 in fact holds for the class of weak bipolarizable graphs [25], a superclass of the bipolarizable graphs.) Corollary 3.1 implies the following result.

Corollary 3.2. *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.*

The algorithm for the recognition of bipolarizable graphs applies Corollary 3.2. 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 Definition 2.1). 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

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 in order to locate the non-common neighbors of u and v;
   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 work similarly as in case 3.2 for each neighbor w of u which is not adjacent to v;
4. Check if the directed subgraph induced by the oriented edges contains a directed cycle; if it does not, print that G is a bipolarizable graph; otherwise, print that it is not.

The correctness of the algorithm follows directly from Corollary 3.2. Observe that for any P_4 $abcd$ of G, the edge bc will be considered in Step 3 of the algorithm, and then the edges ab and cd will be oriented correctly.

Time and Space Complexity. Step 1 takes $O(n+m)$ time since the sorted adjacency lists can be obtained through radix sorting an array of all the ordered pairs of adjacent vertices, while Step 2 takes $O(nm)$ time [23]. Steps 3.2 and 3.3 take constant time per such vertex w; it is assumed that the orientation of an edge is stored in an array of size m for constant-time access and update. For an edge uv, Steps 3.2 and 3.3 is executed $O(deg(u) + deg(v))$ times, where $deg(u)$

denotes the degree of vertex u. Since Step 3.1 also takes $O(deg(u) + deg(v))$ time, Step 3 takes $O\big(\sum_{uv \in E(G)} (deg(u) + deg(v))\big) = O(nm)$ time. Step 4 can be executed by constructing the resulting directed graph and then applying topological sorting on it; if the topological sorting succeeds then no directed cycle exists, otherwise there exists a directed cycle. From this description, it is clear that Step 4 can be completed in $O(n + m)$ time and space. Since the computation of the P_3s participating in P_4s takes linear space, 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 3.1. *Let G be an undirected graph on n vertices and m edges. Then, it can be determined whether G is a bipolarizable graph in $O(nm)$ time and $O(n + m)$ space.*

The recognition algorithm can be used to produce a bipolarizable ordering of the vertices of a bipolarizable graph G. The bipolarizable ordering coincides with the topological ordering of the vertices of the directed graph in Step 4, possibly extended by an arbitrary ordering of any vertices of G which do not participate in the directed graph.

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. In particular, 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 from H whenever the number of such P_3s associated with that vertex 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 $b, c \in H$. In more detail, the algorithm works as follows.

P_4-simplicial Graph Recognition Algorithm

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 $S_0, \ldots, S_{k_v}, S_{k_v+1}$ as described in Definition 2.1, 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 $NumP3[v]$;
3. Collect in a list L the vertices v for which $NumP3[v] = 0$;
4. While the list L is not empty do
   4.1 remove a vertex, say, x, from L;
   4.2 for each vertex u adjacent to x in G do
       if u belongs to H
           traverse the adjacency list $A[u]$ of u and let s_x be the value of the field *set* for the vertex x;
           if $s_x \neq 0$
           then {*there may exist P_3s xuw participating in P_4s of G*}
               for each vertex w in the adjacency list $A[u]$ of u do
                   $s_w \leftarrow$ value of the field *set* for the vertex w;
                   if $w \in H$ and $s_w \neq 0$ and $s_w \neq s_x$
                   then {*xuw is such a P_3 with $x, u, w \in H$*}
                       $NumP3[u] \leftarrow NumP3[u] - 1$;
               if $NumP3[u] = 0$
               then insert u in the list L;
   4.3 remove x from the set H;
5. if the set H is empty, then print that G is a P_4-simplicial graph; otherwise, print that it is not.

To ensure correct execution, the algorithm maintains the following invariant throughout the execution of Step 4 (the proof can be found in [24]).

Invariant 4.1. *At the beginning of every iteration of the while loop in Step 4 of the algorithm, for each vertex v in H, $NumP3[v]$ is equal to the number of P_3s bvc participating in P_4s of G with $b, c \in H$.*

Sketch of the Proof: The proof relies on the fact that $NumP3[v]$ will be decremented precisely once for each P_3 avb participating in a P_4 of G: if a is removed from H before b, then $NumP3[v]$ will be decremented during the removal of a; when b is removed, the P_3 avb will not be considered, even if $v \in H$, because $a \notin H$. ∎

Then, the correctness of the algorithm follows from the correctness of the algorithm of Hoàng and Reed, Property 4.1, and the fact that at any given time the list L contains precisely those vertices that can be removed from H (a vertex x is inserted in L if and only if $NumP3[x] = 0$, i.e., there does not exist any P_3 bxc participating in a P_4 of G with $b, c \in H$).

Time and Space Complexity. The set H can be implemented by means of an array $M[\,]$ of size n, where $M[v] = 1$ if $v \in H$ and 0 otherwise; in this way, insertion, deletion, and membership queries for any vertex of G can be answered in constant time, while the emptiness of H can be checked in $O(n)$ time. Then, Step 1 takes $O(n + m)$ time, Step 4.3 takes $O(1)$ time per vertex removed, and 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 $O\Big(\sum_x \Big(1 + \sum_{u \in N(x)} deg(u)\Big)\Big)$, 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 $O(nm)$. Since the computation of the P_3s participating in P_4s takes linear space, 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.*

5 Class Inclusions and Recognition Time Complexities

Figure 1 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 1 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 the following lemmata (the complete proofs have been omitted due to lack of space but can be found in [24]):

Lemma 5.1. *The class of P_4-simplicial graphs is a proper subset of the class of brittle graphs and a proper superset of the class of weak bipolarizable[1] graphs.*

[1] A graph is weak bipolarizable if it has no induced subgraph isomorphic to C_k ($k \geq 5$), the house graph, or to any of the graphs F_1 and F_2 of Figure 2 [25].

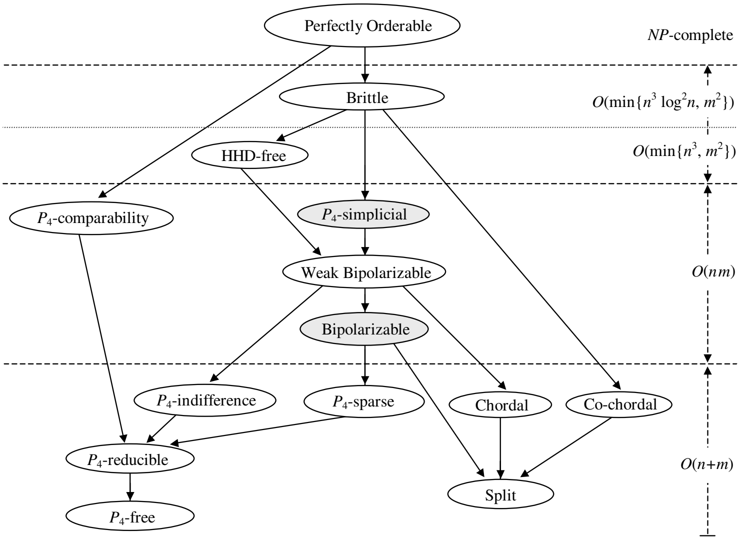

Fig. 1. Class inclusions and recognition time complexities.

Sketch of the Proof: The fact that *P_4-simplicial* $\subseteq$ *Brittle* has been shown in [16]; the subset relation is proper since the graph F_1 of Figure 2 is brittle but not P_4-simplicial. To show that *Weak Bipolarizable* $\subseteq$ *P_4-simplicial*, we apply induction on the size of the graph by taking advantage of Theorem 1 of [25] 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; the proper inclusion follows from the fact that the house graph is P_4-simplicial but not weak bipolarizable. ∎

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.*

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.*

Regarding the relation of P_4-simplicial and the HHD-free and co-chordal graphs, we note that the graph F_1 of Figure 2 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. 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 forbidden subgraph D of [12]) and for the bipolarizable and the P_4-indifference graphs (consider the forbidden subgraphs F_5 of [15] and D of [12]).

Figure 1 also shows the depicted classes of graphs partitioned based on the time complexities of the currently best recognition algorithms: see [7,27] 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, [23,25] for the $O(nm)$-time complexity range, and [10, 19,20,5,26,9,11] 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 [25] 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 $O(nm)$.

6 On Forbidden Subgraphs for P_4-Simplicial Graphs

The minimal set of forbidden subgraphs for the class of bipolarizable graphs has been established in [12,16]. For the class of P_4-simplicial graphs, however, no work on forbidden subgraphs is available in the literature to the best of our knowledge; in this section, we give a number of forbidden subgraphs for this class, and attempt to give a first characterization of them.

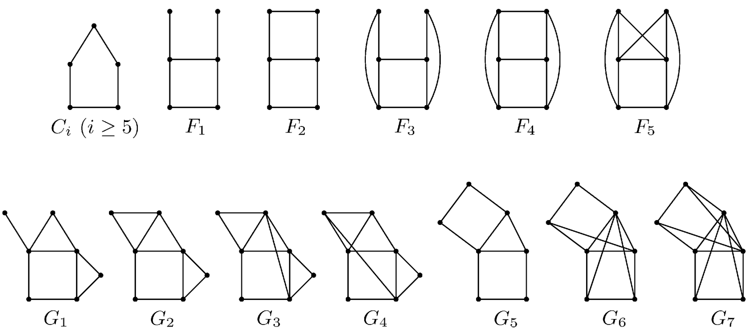

Fig. 2. Some forbidden subgraphs for the class of P_4-simplicial graphs

Clearly, a hole, the graph F_1 (sometimes also called "A"), and the graph F_2 (also known as domino graph or D_6) in Figure 2 are all forbidden subgraphs for P_4-simplicial graphs. On the other hand, the house graph (i.e., $\overline{P}_5$) is P_4-simplicial. Figure 2 shows all forbidden subgraphs on up to 7 vertices; note that F_3 is $\overline{F}_2$, F_4 is $\overline{C}_6$, and F_5 is $\overline{P}_6$. Additionally, even if the holes are excluded, one can easily generate a number of arbitrarily large forbidden subgraphs. Figure 3 gives two such examples.

In any case, Lemma 5.1 implies the following property for all forbidden subgraphs other than a hole, and the graphs F_1 and F_2 of Figure 2:

Lemma 6.1. *Any forbidden subgraph for the class of P_4-simplicial graphs, other than a hole, F_1, and F_2, contains at least one house graph as induced subgraph.*
Following up on Lemma 6.1, we believe that any minimal forbidden subgraph for the class of P_4-simplicial graphs, other than a hole, F_1, and F_2, has at least two houses as induced subgraphs. In fact, we conjecture that the set of such forbidden subgraphs includes a number of graphs containing at least two vertex-sharing houses (see Figure 2) and a small number of graphs that have exactly two vertex-disjoint houses as induced subgraphs (as in Figure 3).

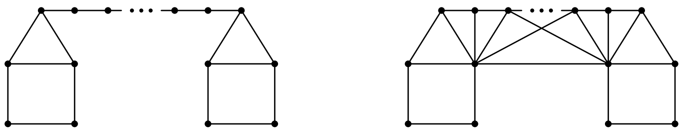

Fig. 3.

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. Our proposed algorithms are simple, use simple data structures and require $O(n + m)$ 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 linear-time recognition of interval graphs. Finally, another interesting open problem is that of completing the 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.
16. C.T. Hoàng and B.A. Reed, Some classes of perfectly orderable graphs, *J. Graph Theory* **13**, 445–463, 1989.
17. C.T. Hoàng and B.A. Reed, P_4-comparability graphs, *Discrete Math.* **74**, 173–200, 1989.
18. C.T. Hoàng and R. Sritharan, Finding houses and holes in graphs, *Theoret. Comput. Sci.* **259**, 233–244, 2001.
19. B. Jamison and S. Olariu, A linear-time recognition algorithm for P_4-sparse graphs, *SIAM J. Comput.* **21**, 381–407, 1992.
20. B. Jamison and S. Olariu, A linear-time algorithm to recognize P_4-reducible graphs, *Theoret. Comput. Sci.* **145**, 329–344, 1995.
21. R.M. McConnell and J. Spinrad, Linear-time modular decomposition and efficient transitive orientation, *Proc. 5th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA'94)*, 536–545, 1994.
22. M. Middendorf and F. Pfeiffer, On the complexity of recognizing perfectly orderable graphs, *Discrete Math.* **80**, 327–333, 1990.
23. S.D. Nikolopoulos and L. Palios, On the recognition of P_4-comparability graphs, *Proc. 28th Workshop on Graph Theoretic Concepts in Computer Science (WG'02)*, LNCS 2573, 355–366, 2002.
24. S.D. Nikolopoulos and L. Palios, Recognizing bipolarizable and P_4-simplicial graphs, *Technical Report 7-03*, Dept of Computer Science, Univ. of Ioannina, 2003.
25. S. Olariu, Weak bipolarizable graphs, *Discrete Math.* **74**, 159–171, 1989.
26. D.J. Rose, R.E. Tarjan, and G.S. Lueker, Algorithmic aspects of vertex elimination on graphs, *SIAM J. Comput.* **5**, 266–283, 1976.
27. A.A. Schäffer, Recognizing brittle graphs: remarks on a paper of Hoàng and Khouzam, *Discrete Appl. Math.* **31**, 29–35, 1991.