Minimal Separators and Perfect Matchings in P_4-sparse 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: In this paper, we consider problems pertaining to the minimal separators and perfect matchings in P_4-sparse graphs. Specifically, we determine the minimal separators of P_4-sparse graphs and establish bounds on their number. We show that a P_4-sparse graph G on n vertices and m edges has fewer than $2n/3$ minimal separators of total description size at most $4m/3$. The bound on the number of minimal separators is tight and is also tight for the class of complement reducible graphs or cographs, a well-known subclass of P_4-sparse graphs. Our results enable us to present a linear-time and linear-space algorithm for computing the number of minimal separators of a given P_4-sparse graph; the algorithm can be modified to report the minimal separators of the input graph in linear time and space as well.

We also address the problem of computing a maximum-size subgraph of a P_4-sparse graph which admitts a perfect matching; in the case where the graph has a perfect matching, the solution to the problem is the entire graph. We establish a characterization of such subgraphs, and describe an algorithm for the problem which, for a P_4-sparse graph on n vertices and m edges, runs in $O(n+m)$ time and space. These results also hold for the class of cographs.

Keywords: Minimal separation, maximum-size subgraphs, maximum matchings, perfect matching, P_4-sparse graphs, cographs, perfect graphs.

1 Introduction

The class of P_4-*sparse* graphs was introduced by Hoàng in his doctoral dissertation [18] as the class of graphs for which every set of five vertices induces at most one P_4 (chordless path on four vertices). Hoàng gave a number of characterizations for these graphs and showed that the P_4-sparse graphs are perfect in the sense of Berge (a graph G is perfect if for every induced subgraph H of G, the chromatic number of H equals the clique number of H), and in fact perfectly orderable in the sense of Chvátal; see [4, 15].

The study of P_4-sparse graphs led naturally to constructive characterizations that implied several linear-time recognition algorithms; such an algorithm was given in [21]. Additionally, optimization problems, such as, the coloring, the minimum clique cover, the maximum weighted clique and stable set, and the minimum fill-in, can be solved in linear time as well [23, 14]. The class of P_4-sparse graphs generalizes both the *complement reducible* graphs, also known as *cographs*, and the P_4-*reducible* graphs: the well known class of cographs was introduced in the early 1970s by Lerchs [24] as the class of graphs for which no induced subgraph is isomorphic to a P_4; Jamison and Olariu [20] introduced the class of P_4-reducible graphs as those in which no vertex belongs to more than one induced P_4. Both cographs and P_4-reducible graphs can be recognized in linear time [5, 8, 20].

The class of P_4-sparse, and also the classes of cographs and P_4-reducible graphs, have been studied extensively in recent years and find applications in many areas of applied mathematics, computer science, and engineering, which involve graphs that feature "local density" and also nice algorithmic and structural properties. Indeed, the structure of the P_4-sparse graphs incorporates such local density properties since they are graphs with few P_4s; we note that researchers tend to equate the notion of local density with the absence of P_4s.

A subset S of vertices of a graph G is called a *separator* if the subgraph $G[V(G) - S]$ of G induced by $V(G) - S$ is not connected; note that the definition implies that the empty set $\emptyset$ is a separator of every disconnected graph and of no connected graph. A separator S of G is called an *ab-separator* if the vertices a and b of G belong to different connected components in $G[V(G)-S]$. A separator S is called a *minimal ab-separator* if it is an ab-separator and no proper subset of S is an ab-separator, and S is called a *minimal separator* if it is a minimal ab-separator for a pair $\{a, b\}$ of vertices of G [3, 2]; again, the way we defined the separators of a graph implies that $\emptyset$ is a minimal separator of every disconnected graph. We also mention that, in general, minimal separators of a graph can be generated by computing the neighborhoods of the connected components of subgraphs resulting after the removal of certain vertex sets [3].

The interest in minimal separators comes both from the fact that the minimal separators are used as a structural tool in several algorithmic applications and from their close relationship to the extremities of a graph [1]; detecting and using extremities is a way (although not the only one) of dealing with graph problems such as searching, recognition, and decomposition, since they allow only "local manipulations", which in many cases ensure efficiency. For many graph classes, recognition and several optimization problems were solved by appropriate processing of the graphs' extremities: [1] mentions the work of Roberts who used extremities to characterize proper interval graphs [27], of Golumbic and Gross who defined an edge-elimination scheme for chordal bipartite graphs [16], of Fulkerson and Gross who introduced an elimination scheme which they used to characterize chordal graphs with an associated greedy recognition algorithm [13], of Dahlhaus *et al.* who extended elimination schemes to domination graphs [11], and of Corneil *et al.* who showed that strongly chordal graphs have strongly simplicial vertices and that AT-graphs have a domination path with extremities at each end [7].

A *matching* M of a graph G is a subset of the edge set $E(G)$ such that no two edges in M share a common endpoint; M is a *maximum matching* if it contains a maximum number of edges; M is a *perfect matching* if every vertex of G is an endpoint of an edge in M. The best known algorithm for solving the maximum matching problem in general graphs is due to Micali and Vazirani [26] and has $O(\sqrt{n}\, m)$ time complexity; note that in P_4-sparse graphs and cographs, the same problem is solved in linear time due to the algorithm of Fouquet, Parfenoff, and Thuillier [12].

In this paper, we determine the minimal separators of P_4-sparse graphs, and show that such a graph G has fewer than $2n/3$ minimal separators whose total description size does not exceed $4m/3$, where n is the number of vertices and m the number of edges of G; in particular, if G is disconnected, the number of its minimal separators is at most $(2n-1)/3$, whereas if it is connected, it is at most $(2n-2)/3$. We also show that these bounds are tight by exhibiting a family of P_4-sparse graphs which have that many separators; these graphs are cographs which shows that these lower bounds apply for the class of cographs as well. Based on our results and by using the modular decomposition tree representation of a P_4-sparse graph, we also describe a linear-time algorithm for the following problem:

- Problem 1: Given a P_4-sparse graph G, find the number of all the minimal separators of G.

The algorithm can be easily modified to report all the minimal separators of the graph G in time linear in the size of G. Our results enable us to include the class of P_4-sparse graphs, and also the classes of cographs and P_4-reducible graphs, to the collection of graph classes which have a linear number of minimal separators, as are the chordal graphs and the weakly chordal graphs; we note that the number of minimal separators in a chordal graph is at most n [28], while the corresponding number in a weakly chordal graph is at most $n + m$ [2].

We also address the following optimization problem on P_4-sparse graphs:

- Problem 2: Given a P_4-sparse graph G, find a maximum-size subgraph of G which has a perfect matching.

In other words, we want to find and remove the smallest number of edges so that the graph which we obtain if we ignore any isolated vertices has a perfect matching. The problem belongs to the family of problems in which we are asked to remove as few edges or vertices as possible so that the resulting subgraph has some particular properties.

We show that any maximum-size subgraph of a graph G which has a perfect matching is a subgraph induced by the vertices of a maximum matching of G. In this way, we reduce the problem to that of finding a maximum-size subgraph induced by the vertices of a maximum matching of G. Then, we establish a characterization of such subgraphs which by means of the modular decomposition tree representation of a P_4-sparse graph enables us to obtain a linear-time solution.

The paper is organized as follows. In Section 2, we present the notation, related terminology, and background results which we need for our algorithms. In Section 3, we present our results on minimal separators of P_4-sparse graphs, and give our algorithm for Problem 1. Our results and the solution for Problem 2 are described in Section 4. Finally, Section 5 concludes the paper with a summary of our results and possible extensions.

2 Theoretical Framework

We consider finite undirected graphs with no loops or multiple edges. For a graph G, we denote its vertex and edge set by $V(G)$ and $E(G)$, respectively. Let S be a subset of the vertex set of a graph G. Then, the subgraph of G induced by S is denoted by $G[S]$.

A *path* in the graph G is a sequence of vertices $v_0 v_1 \cdots v_k$ such that $v_i v_{i+1} \in E(G)$, $0 \leq i \leq k-1$; 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 no vertex occurs more than once; it is called *trivial* if its length is equal to 0. A simple path $v_0 v_1 \cdots v_k$ is *chordless* if $v_i v_j \notin E(G)$ for any two non-consecutive vertices v_i, v_j in the path. Throughout the paper, the chordless path on k vertices is denoted by P_k; in particular, a chordless path on 4 vertices is denoted by P_4. If the graph G contains a path from a vertex x to a vertex y, we say that *x is connected to y*. The graph G is *connected* if x is connected to y for every pair of vertices $x, y \in V(G)$. The *connected components* (or *components*) of G are the equivalence classes of the "is connected to" relation on the vertex set $V(G)$ of G. The *co-connected components* (or *co-components*) of G are the connected components of the complement $\overline{G}$ of the graph G.

The *neighborhood* $N(x)$ of a vertex x of the graph G is the set of all the vertices of G which are adjacent to x. The *closed neighborhood* of x is defined as $N[x] := N(x) \cup \{x\}$. The *degree*

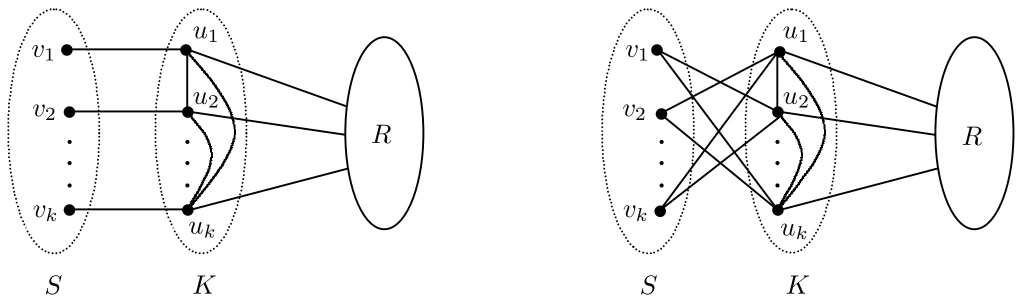


Figure 1: A thin and a thick spider.

of a vertex x in a graph G, denoted $degree(x)$, is the number of edges incident on x; thus, $degree(x) = |N(x)|$. A *clique* is a set of pairwise adjacent vertices; a *stable set* is a set of pairwise non-adjacent vertices.

2.1 P_4-sparse graphs

Below, we review properties and characterizations of P_4-sparse graphs and prove results which we use in our algorithm for the solution of the problem of finding the minimal separators of a P_4-sparse graph.

A graph G is called a *spider* if the vertex set $V(G)$ of the graph G admits a partition into sets S, K, and R such that:

S1: $|S| = |K| \geq 2$, the set S is a stable set, and the set K is a clique;

S2: all the vertices in R are adjacent to all the vertices in K and to no vertex in S;

S3: there exists a bijection $f : S \to K$ such that exactly one of the following statements holds:

(i) for each vertex $v \in S$, $N(v) \cap K = \{f(v)\}$;

(ii) for each vertex $v \in S$, $N(v) \cap K = K - \{f(v)\}$.

The triple (S, K, R) is called the *spider-partition.* A graph G is a *prime spider* if G is a spider with $|R| \leq 1$. If the condition of case S3(i) holds then the spider G is called a *thin spider*, whereas if the condition of case S3(ii) holds then G is a *thick spider*; note that the complement of a thin spider is a thick spider and vice versa. Figure 1 depicts two spiders with partition sets $S = \{v_1, v_2, \ldots, v_k\}$, $K = \{u_1, u_2, \ldots, u_k\}$, and R: the spider on the left is a thin spider with $N(v_i) \cap K = \{u_i\}$, whereas the one on the right is a thick spider with $N(v_i) \cap K = K - \{u_i\}$, for every $v_i \in S$.

Observation 2.1 (Observation 2.8 in [22]). *If a graph G is a spider, then exactly one of the following statements holds:*

(i) for every $v \in S$ and $u \in K$, $degree(v) = 1$ and $degree(u) = |V(G)| - |S|$;

(ii) for every $v \in S$ and $u \in K$, $degree(v) = |K| - 1$ and $degree(u) = |V(G)| - 2$.

Again, items (i) and (ii) correspond to the case of a thin spider and a thick spider, respectively.

Observation 2.2 (Observation 2.9 in [22]). *If a graph G is a spider with partition (S, K, R) and R is nonempty, then for every $v \in S$, $u \in K$, and $r \in R$, $degree(v) < degree(r) < degree(u)$.*

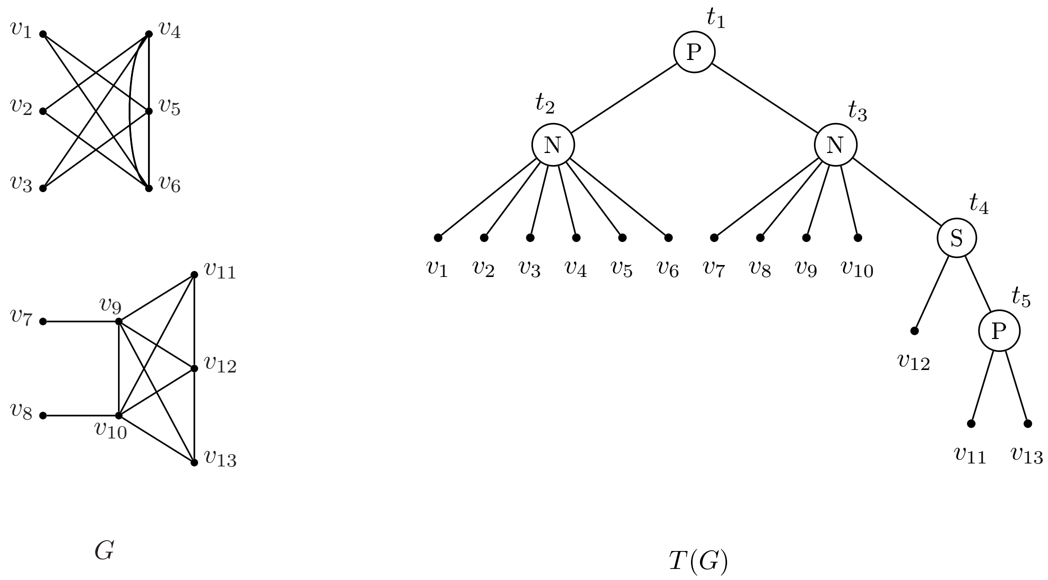


Figure 2: A disconnected P_4-sparse graph G on 13 vertices and the corresponding modular decomposition tree $T(G)$.

It is not difficult to see that a spider with $|K| = |S| = k$ contains exactly $\frac{k(k-1)}{2} + k'$ P_4s, where k' is the number of P_4s in the subgraph $G[R]$. From the definition of the spider and Observations 2.1 and 2.2, it follows that if G is a spider, then the sets S, K, and R are unique (see also [22]). Finally, from the properties of a spider G, and also from the definition of the P_4-sparse graphs, it easily follows that G is P_4-sparse iff the graph $G[R]$ is P_4-sparse.

Let us now return to general P_4-sparse graphs. Then, the following result holds:

Lemma 2.1 (Theorem 2.13 in [22]). *For a graph G, the following conditions are equivalent:*

(i) G is a P_4-sparse graph;

(ii) for every induced subgraph H of G with at least two vertices, exactly one of the following statements is satisfied:

(ii.1) H is disconnected;

(ii.2) $\overline{H}$ is disconnected;

(ii.3) H is a spider.

2.2 Modular decomposition

A subset M of vertices of a graph G is said to be a *module* of G, if every vertex outside M is either adjacent to all vertices in M or to none of them. The empty set, the singletons, and the vertex set $V(G)$ are *trivial* modules and whenever G has only trivial modules it is called a *prime* (or *indecomposable*) *graph*. A non-trivial module is also called *homogeneous set*. A module M of the graph G is called a *strong module*, if for any module M' of G, either $M' \cap M = \emptyset$ or one module is included into the other. Furthermore, a module in G is also a module in $\overline{G}$.

The *modular decomposition* of a graph G is a linear-space representation of all the partitions of $V(G)$ where each partition class is a module. The *modular decomposition tree* $T(G)$ of the graph G (or *md-tree* for short) is a unique labeled tree associated with the modular decomposition of G in which the leaves of $T(G)$ are the vertices of G and the set of vertices of G which are the leaves of the subtree rooted at an internal node induces a strong module of G. Thus, the md-tree $T(G)$ represents all the strong modules of G. An internal node is labeled by either P (for *parallel* module), S (for *series* module), or N (for *neighborhood* module). The module corresponding to a P-node induces a disconnected subgraph of G, that of an S-node induces a connected subgraph of G whose complement is a disconnected subgraph, and that of an N-node induces a connected subgraph of G whose complement is also a connected subgraph. Note that if the md-tree $T(G)$ does not contain any N-node then it is precisely the special case of the cotree and G is a cograph, since it has only parallel and series modules.

The process of constructing the md-tree $T(G)$ of the graph G is defined as follows: parallel modules are decomposed into their connected components, series modules into their co-connected components, and neighborhood modules into their maximal strong submodules. The efficient construction of the md-tree $T(G)$ has received a great deal of attention. It can be shown that for every graph G the tree $T(G)$ is unique up to isomorphism and it can be constructed in linear time; the first linear-time algorithms for the construction of the md-tree are described in [9, 25], while more recent and more practical ones can be found in [10, 17]. In Figure 2 we show an example of a disconnected P_4-sparse graph and its md-tree.

Let t be an internal node of the md-tree $T(G)$ of a graph G. We denote by $M(t)$ the module corresponding to t which consists of the set of vertices of G associated with the subtree of $T(G)$ rooted at node t; note that $M(t)$ is a strong module for every (internal or leaf) node t of $T(G)$. Let $\{a_1, a_2, \dots, a_p\}$ be the set of children of node t in $T(G)$. We denote by $G(t)$ the *representative graph* of the module $M(t)$ defined as follows:

- ∘ $V(G(t)) = \{a_1, a_2, \dots, a_p\}$, and
- ∘ $E(G(t)) = \{a_i a_j \mid v_k v_\ell \in E(G),\ v_k \in M(a_i) \text{ and } v_\ell \in M(a_j)\}$.

Note that by the definition of a module, if a vertex of $M(a_i)$ is adjacent to a vertex of $M(a_j)$ then every vertex of $M(a_i)$ is adjacent to every vertex of $M(a_j)$. Thus $G(t)$ is isomorphic to the graph induced by a subset of $M(t)$ consisting of a single vertex from each maximal strong submodule of $M(t)$ in the modular decomposition of G. It is easy to show that the following lemma holds (see also [14]):

Lemma 2.2. *Let G be a graph, $T(G)$ its modular decomposition tree, and t an internal node of $T(G)$. Then, $G(t)$ is an edgeless graph if t is a P-node, $G(t)$ is a complete graph if t is an S-node, and $G(t)$ is a prime graph if t is an N-node.*

In particular, for the case of P_4-sparse graphs, Giakoumakis and Vanherpe [14] showed the following result:

Lemma 2.3. *Let G be a graph and let $T(G)$ be its modular decomposition tree. The graph G is P_4-sparse iff for every N-node t of $T(G)$, $G(t)$ is a prime spider with a spider-partition (S, K, R) and no vertex in $S \cup K$ is an internal node of $T(G)$.*

(Recall that the graph $G(t)$ has as vertices the children of the node t of $T(G)$).) The above lemma implies that every N-node t of the md-tree $T(G)$ of a P_4-sparse graph G has either $2k$ or $2k+1$ children, where $|S| = |K| = k \geq 2$ and $|R| \leq 1$; the sets S, K, and R form the spider-partition of the graph $G(t)$. More precisely, the N-node t has k children which correspond to S, k children which correspond to K, and either no other child if $R = \emptyset$ or one more child if $R \neq \emptyset$ (in this case, $|R| = 1$ and this child is the root of a subtree of $T(G)$). The children which correspond to S and K are leaves in $T(G)$ and, thus, they are vertices of G, while the child which corresponds to R, if $R \neq \emptyset$, is either a leaf (i.e., a vertex of G) or an internal node labeled by either P, S, or N. The md-tree $T(G)$ depicted in Figure 2 contains two N-nodes, the nodes t_2 and t_3. The graph $G(t_2)$ is a prime spider on 6 vertices with spider-partition $S = \{v_1, v_2, v_3\}$, $K = \{v_4, v_5, v_6\}$, and $R = \emptyset$, while $G(t_3)$ is a prime spider on 5 vertices with spider-partition $S = \{v_7, v_8\}$, $K = \{v_9, v_{10}\}$, and $R = \{t_4\}$; the internal node t_4 is an S-node. The graph $G[M(t_2)]$ is a spider (prime spider) with $R = \emptyset$, and the graph $G[M(t_3)]$ is also a spider (non prime spider) with $R = \{v_{11}, v_{12}, v_{13}\}$.

Lemma 2.3 also implies a linear-time algorithm for the recognition problem, and linear-time algorithms for classical optimization problems on P_4-sparse graphs [14].

3 Minimal Separators

In this section, we prove theoretical results on minimal separators for the class of P_4-sparse graphs, provide tight bounds on their number, and describe a linear-time algorithm for counting the number of minimal separators of a given P_4-sparse graph. Recall that we consider that the empty set is a minimal separator of every disconnected graph (this works well for our purposes; see Remark 3.1); if this is not desired, then for a disconnected graph, ignore the empty set from the set of minimal separators given below and subtract 1 from their number.

Lemma 3.1. *Let G be a P_4-sparse graph. Then,*

(i) G is disconnected: if $C_1, C_2, \ldots, C_p$, $p \geq 2$, are the connected components of G and $\{\sigma_{i,1}, \sigma_{i,2}, \ldots, \sigma_{i,\ell_i}\}$ are the sets of minimal separators of the subgraph $G[C_i]$, $1 \leq i \leq p$, respectively, then the minimal separators of G are

- *the empty set $\emptyset$ and*
- *the separators $\sigma_{i,j}$ for all $1 \leq i \leq p$ and $1 \leq j \leq \ell_i$.*

(ii) $\overline{G}$ is disconnected: if $C'_1, C'_2, \ldots, C'_q$, $q \geq 2$, are the co-components of G and $\{\sigma'_{i,1}, \sigma'_{i,2}, \ldots, \sigma'_{i,\ell'_i}\}$ are the sets of minimal separators of the subgraph $G[C'_i]$, $1 \leq i \leq q$, respectively, then the minimal separators of G are

- *the sets $\sigma'_{i,j} \cup (V(G) - C'_i)$ for all $1 \leq i \leq q$ and $1 \leq j \leq \ell'_i$.*

(iii) G is a spider: if (S, K, R) is the spider partition of G and $\{\sigma_1, \sigma_2, \ldots, \sigma_\ell\}$ is the set of minimal separators of the subgraph $G[R]$, then the minimal separators of G are

- *the sets $\sigma_i \cup K$, for each separator σ_i, $1 \leq i \leq \ell$, of $G[R]$ and*
- *the subsets of K of size 1 if G is a thin spider or the subsets of K of size $|K| - 1$ if G is a thick spider.*

Proof: First, observe that cases (i)-(iii) cover all possibilities (Lemma 2.1).

(i) Since the graph G is disconnected, the empty set is a separator; it is important to note that $\sigma_{i,j} \neq \emptyset$ since the subgraphs $G[C_1], G[C_2], \ldots, G[C_p]$ are connected. In order to determine the minimal separators of G, we need to examine pairs of non-adjacent vertices of G. Clearly, we need only consider pairs of vertices that both belong to the same connected component C_i of G. But this implies that the sought separators are precisely the separators $\sigma_{i,j}$.

(ii) We examine pairs of vertices of G to determine their minimal separators. Since any two vertices belonging to different co-components of G are adjacent in G, we need only consider pairs of vertices that both belong to the same co-component C'_i; any two such vertices are adjacent in G iff they are adjacent in the induced subgraph $G[C'_i]$. Then, it is not difficult to see that the set $\sigma'_{i,j} \cup (V(G) - C'_i)$ is an xy-separator in G for two vertices $x, y \in C'_i$ iff $\sigma'_{i,j}$ is an xy-separator in $G[C'_i]$. Finally, the minimality of $\sigma'_{i,j} \cup (V(G) - C'_i)$ follows from the minimality of the separator $\sigma'_{i,j}$ in $G[C'_i]$ and the fact that any vertex in $V(G) - C'_i$ is adjacent to all vertices in C'_i.

(iii) In order to determine the minimal separators, we need to consider pairs of vertices x, y such that either $x \in S$ and $y \in R \cup (K - N(x)) \cup (S - \{x\})$, or $x, y \in R$. In the latter case, the set $\sigma_i \cup K$ is an xy-separator of G iff σ_i is an xy-separator of $G[R]$; additionally, the minimality of $\sigma_i \cup K$ follows from the minimality of σ_i and the fact that all vertices in K are adjacent to all vertices in R. For the former case, we distinguish cases depending on whether the graph G is a thin or a thick spider. Suppose that G is a thin spider and let $f(v) \in K$ be the only neighbor of vertex $v \in S$. Consider a vertex $x \in S$. Then, for any vertex $y \in R \cup (K - \{f(x)\})$, the minimal xy-separator is $\{f(x)\}$. If $y \in S - \{x\}$, then the removal of $f(x)$ or $f(y)$ suffices to disconnect x from y so that the minimal separators are $\{f(x)\}$ and $\{f(y)\}$. Thus, by taking into account all vertices in S, we obtain as minimal separators all the subsets of the set K of size 1. In a similar fashion, we can show that if G is a thick spider, the minimal xy-separators of G when $x \in S$ are all the subsets of the set K of size $|K| - 1$. ∎

Remark 3.1: It is important to note that in case (ii) of Lemma 3.1 if a subgraph $G[C'_i]$ is disconnected, the set $V(G) - C'_i$ is a minimal separator; this separator is contained in the separators described in Lemma 3.1(ii) because $G[C'_i]$ contributes the minimal separator $\emptyset$. Additionally, in case (iii), if $G[R]$ is disconnected, its minimal separator $\emptyset$ enables us to correctly include the set K in the set of minimal separators of G. On the other hand, no separator of a connected P_4-sparse graph is equal to $\emptyset$; in case (ii), $V(G) - C'_i \neq \emptyset$, while in case (iii), $K \neq \emptyset$.

Lemma 3.1 implies the following theorem (note that the total description size of a set of separators is equal to the sum of the sizes of the separators in the set).

Theorem 3.1. *Let G be a P_4-sparse graph on n vertices and m edges. Then, if G is disconnected it has at most $(2n-1)/3$ minimal separators, whereas if it is connected it has at most $(2n-2)/3$ minimal separators. In either case, the total description size of the minimal separators does not exceed $4m/3$.*

Proof: We use induction on the number n of vertices of the graph G (note that there exist P_4-sparse graphs on any number of vertices). For the basis step, we consider a graph on $n = 1$ vertex and $m = 0$ edges: the graph is P_4-sparse, is connected, and has no separators; since $0 \leq (2 \cdot 1 - 2)/3$ and $0 \leq 4 \cdot 0/3$, the basis case holds. For the inductive hypothesis, we assume that the theorem holds for all P_4-sparse graphs on at most $n - 1 \geq 1$ vertices. The inductive step requires us to show that the theorem holds for all P_4-sparse graphs on n vertices. Let us consider such a graph G and let m be the number of its edges. Then, by Lemma 2.1, exactly one of the following three cases holds:

(i) *G is disconnected:* Let $C_1, C_2, \ldots, C_p$, $p \geq 2$, be the connected components of G. If $n_1, n_2, \ldots, n_p$ and $m_1, m_2, \ldots, m_p$ are the numbers of vertices and edges of the subgraphs $G[C_1], G[C_2], \ldots, G[C_p]$ respectively, then

$$\begin{aligned} n &= n_1 + n_2 + \ldots + n_p && (1) \\ m &= m_1 + m_2 + \ldots + m_p && (2) \end{aligned}$$

while in accordance with Lemma 3.1(i) we have

$$\begin{aligned} \text{number of min-separators of } G &= 1 + \sum_{i=1}^{p} (\text{number of min-separators of } G[C_i]) && (3) \\ \text{size of min-separators of } G &= \sum_{i=1}^{p} (\text{size of min-separators of } G[C_i]). && (4) \end{aligned}$$

Additionally, the subgraphs $G[C_1], G[C_2], \ldots, G[C_p]$ are P_4-sparse graphs on at most $n-1$ vertices each, and thus the inductive hypothesis applies on each of them; since they are connected, it holds that

$$\begin{aligned} \text{number of min-separators of } G[C_i] &\leq \frac{2 \cdot n_i - 2}{3} && (5) \\ \text{size of min-separators of } G[C_i] &\leq \frac{4 \cdot m_i}{3}. && (6) \end{aligned}$$

Combining the inequality (5) with Eqs. (3) and (1), we have

$$\text{number of min-separators of } G \;\leq\; 1 + \sum_{i=1}^{p} \frac{2 \cdot n_i - 2}{3} \;=\; 1 + \frac{2 \cdot n}{3} - \frac{2p}{3}.$$

Since $p \geq 2$, we have $1 - 2p/3 \leq -1/3$, which implies that

$$\text{number of min-separators of } G \;\leq\; \frac{2 \cdot n - 1}{3}. \tag{7}$$

Moreover, Eq. (4) with Eqs. (6) and (2) give

$$\text{size of min-separators of } G \;\leq\; \sum_{i=1}^{p} \frac{4 \cdot m_i}{3} \;=\; \frac{4 \cdot m}{3}. \tag{8}$$

Inequalities (7) and (8) establish the desired bounds for a disconnected P_4-sparse graph on n vertices and m edges.

(ii) *$\overline{G}$ is disconnected:* Let $C'_1, C'_2, \ldots, C'_q$, $q \geq 2$, be the co-components of the graph G and let $n'_1, n'_2, \ldots, n'_q$ and $m'_1, m'_2, \ldots, m'_q$ be the numbers of vertices and edges of the subgraphs $G[C'_1], G[C'_2], \ldots, G[C'_q]$ respectively. We work as in the previous case and we have

$$\begin{aligned} n &= n'_1 + n'_2 + \ldots + n'_q && (9) \\ m &= m'_1 + m'_2 + \ldots + m'_q + \frac{1}{2} \sum_{i=1}^{q} n'_i \cdot (n - n'_i). && (10) \end{aligned}$$

Eq. (10) comes from the fact that, in addition to the edges in the subgraphs $G[C'_1], \ldots, G[C'_q]$, G contains edges between any pair of vertices belonging to different co-components of G.

Additionally, Lemma 3.1(ii) implies

$$\begin{aligned}
\text{number of min-separators of } G &= \sum_{i=1}^{q} (\text{number of min-separators of } G[C'_i]) && (11)\\
\text{size of min-separators of } G &= \sum_{i=1}^{q} \sum_{j=1}^{\ell'_i} \big(|\sigma'_{i,j}| + (n - n'_i) \big) && (12)
\end{aligned}$$

where $\sigma'_{i,j}$ $(1 \leq j \leq \ell'_i)$ are the minimal separators of $G[C'_i]$; since $\sum_{j=1}^{\ell'_i} |\sigma'_{i,j}|$ is precisely the total size of the minimal separators of $G[C'_i]$, Eq. (12) is rewritten

$$\text{size of min-separators of } G = \sum_{i=1}^{q} \Big((\text{size of min-separators of } G[C'_i]) + \ell'_i \cdot (n - n'_i) \Big). \quad (13)$$

The subgraphs $G[C'_1], G[C'_2], \ldots , G[C'_q]$ are P_4-sparse graphs on at most $n-1$ vertices each, and thus the inductive hypothesis applies on each of them; they may be connected or disconnected, and hence

$$\text{number } \ell'_i \text{ of min-separators of } G[C'_i] \leq \frac{2 \cdot n'_i - 1}{3} \quad (14)$$

since $(2 \cdot n'_i - 2)/3 \leq (2 \cdot n'_i - 1)/3$, and

$$\text{size of min-separators of } G[C'_i] \leq \frac{4 \cdot m'_i}{3}. \quad (15)$$

Then, from Eq. (11) in conjunction with Eqs. (14) and (9) and the fact that $q \geq 2$, we have

$$\text{number of min-separators of } G \leq \sum_{i=1}^{q} \frac{2 \cdot n'_i - 1}{3} = \frac{2 \cdot n}{3} - \frac{q}{3} \leq \frac{2 \cdot n - 2}{3}. \quad (16)$$

Additionally, Eqs. (13) and (15) imply

$$\text{size of min-separators of } G \leq \sum_{i=1}^{q} \Big(\frac{4 \cdot m'_i}{3} + \ell'_i \cdot (n - n'_i) \Big)$$

which in light of Eqs. (14) and (10) gives

$$\text{size of min-separators of } G \leq \sum_{i=1}^{q} \Big(\frac{4 \cdot m'_i}{3} + \frac{2 \cdot n'_i - 1}{3} \cdot (n - n'_i) \Big) \leq \frac{4 \cdot m}{3}. \quad (17)$$

Inequalities (16) and (17) establish the desired bounds for a connected P_4-sparse graph on n vertices and m edges.

(iii) *G is a spider:* Let (S, K, R) be the spider partition of G and let $|S| = |K| = k \geq 2$, which implies that $|R| = n - 2k$. Lemma 3.1(iii) implies that

$$\begin{aligned}
\text{number of min-separators of } G &= k + (\text{number of min-separators of } G[R]) && (18)\\
\text{size of min-separators of } G &= \sum_{i=1}^{\ell} \big(|\sigma_i| + k \big) + \begin{cases} k & \text{if } G \text{ is a thin spider} \\ k \cdot (k-1) & \text{if } G \text{ is a thick spider} \end{cases} && (19)
\end{aligned}$$

where σ_i $(1 \le i \le \ell)$ are the minimal separators of $G[R]$. Since the subgraph $G[R]$ is P_4-sparse on at most $n-1$ vertices, the inductive hypothesis applies on it; as it may be connected or disconnected, we have

$$\text{number } \ell \text{ of min-separators of } G[R] \;\le\; \frac{2 \cdot |R| - 1}{3} \;=\; \frac{2 \cdot (n-2k) - 1}{3} \tag{20}$$

$$\text{size of min-separators of } G[R] \;=\; \sum_{i=1}^{\ell} |\sigma_i| \;\le\; \frac{4 \cdot m_R}{3} \tag{21}$$

where m_R is the number of edges of $G[R]$. Eqs. (18) and (20) and the fact that $k \ge 2$ imply that

$$\text{number of min-separators of } G \;\le\; k + \frac{2 \cdot (n-2k) - 1}{3} \;=\; \frac{2 \cdot n}{3} - \frac{k+1}{3} \;\le\; \frac{2 \cdot n - 2}{3} \tag{22}$$

as desired for a connected P_4-sparse graph on n vertices. Since for the number m of edges of the graph G it holds that

$$m \;=\; m_R + k \cdot |R| + k \cdot (k-1)/2 + \begin{cases} k & \text{if } G \text{ is a thin spider} \\ k \cdot (k-1) & \text{if } G \text{ is a thick spider} \end{cases} \tag{23}$$

and $\sum_{i=1}^{\ell} k = \ell \cdot k \le 2 \cdot k \cdot |R|/3$ (due to Eq. (20)), Eq. (19) in conjunction with Eqs. (21) and (23) gives

$$\text{size of min-separators of } G \;\le\; \frac{4 \cdot m}{3}.$$

The inductive proof is complete since, in each case, the statement of the theorem holds for the graph on n vertices. Therefore, the theorem holds for all P_4-sparse graphs. ∎

In fact, we can show that the above bounds on the number of minimal separators of a P_4-sparse graph are tight. Consider the following two families of graphs:

F_0 is a graph on a single vertex and no edges

and for each $i \ge 1$:

H_i is a disconnected graph with 2 components, each being a copy of the graph F_{i-1},

F_i is a connected graph with 2 co-components, each being a copy of the graph H_i.

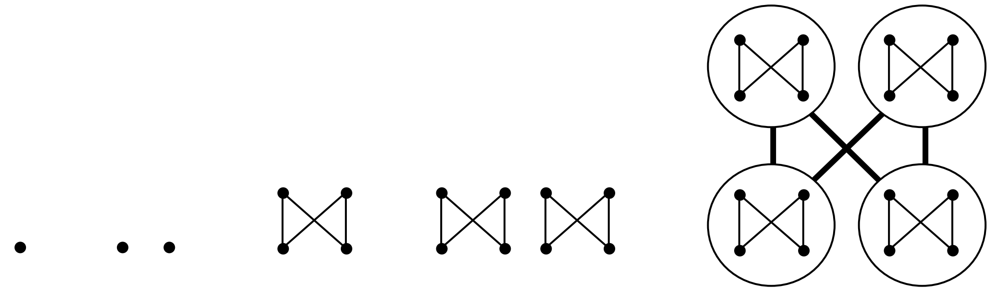
(Figure 3 depicts the graphs F_i and H_i for $i \le 2$; the heavy line connecting a pair of "blobs" in F_2 indicates that every vertex in one of these blobs is adjacent to every vertex in the other blob.) It is not difficult to see that the graphs F_i and H_i have the following properties:

P1: each of the graphs F_i $(i \ge 0)$ and H_i $(i \ge 1)$ is a cograph and hence a P_4-sparse graph;

P2: the number of vertices of the graph F_i is $|V(F_i)| = 4^i$, $i \ge 0$;

P3: the number of vertices of the graph H_i is $|V(H_i)| = 2 \cdot 4^{i-1}$, $i \ge 1$.

Then, we can show the following result:



F_0 H_1 F_1 H_2 F_2

Figure 3: The graphs F_0, H_1, F_1, H_2, and F_2.

Lemma 3.2. *The number of minimal separators of the graphs F_i ($i \geq 0$) and H_i ($i \geq 1$) are $(2 \cdot |V(F_i)| - 2)/3$ and $(2 \cdot |V(H_i)| - 1)/3$ respectively.*

Proof: In light of the definition of the graphs and Lemma 3.1(i) and (ii), we observe that

$$\text{number of min-separators of } H_i \;=\; 1 + 2 \cdot (\text{number of min-separators of } F_{i-1}); \tag{24}$$

$$\text{number of min-separators of } F_i \;=\; 2 \cdot (\text{number of min-separators of } H_i). \tag{25}$$

Eqs. (24) and (25) imply that

$$\text{number of min-separators of } F_i \;=\; 2 + 4 \cdot (\text{number of min-separators of } F_{i-1}).$$

This equality and the fact that $|V(F_i)| = 4^i$ (see Property P2) make it easy to show by induction on i ($i \geq 0$) that

$$\text{number of min-separators of } F_i \;=\; \frac{2 \cdot |V(F_i)| - 2}{3}.$$

Then, from Eq. (24) and because $|V(H_i)| = 2 \cdot |V(F_{i-1})|$ (see Properties P2 and P3), it follows that

$$\text{number of min-separators of } H_i \;=\; 1 + 2 \cdot \frac{2 \cdot |V(F_{i-1})| - 2}{3} \;=\; \frac{2 \cdot |V(H_i)| - 1}{3}. \qquad \blacksquare$$

Since the graphs F_i and H_i are cographs (Property P1), the lower bounds established in Lemma 3.2 for the number of minimal separators also apply for the class of cographs.

Finally, we describe an algorithm which computes the number of minimal separators of a given P_4-sparse graph G. The algorithm relies on Lemma 3.1.

Algorithm MinSepNum

Input: a P_4-sparse graph G.

Output: the number of minimal separators of G.

1. Construct the modular decomposition tree $T(G)$ of the input graph G;
2. Execute the subroutine *process*(*root*), where *root* is the root node of the tree $T(G)$; the number of minimal separators of G is the number returned by the subroutine.

where the description of the subroutine *process*( ) is as follows:

process(node t)

Input: node t of the modular decomposition tree $T(G)$ of the input graph G.

Output: the number of minimal separators of the subgraph H of G corresponding to the subtree of $T(G)$ rooted at node t.

1. `if` t is a leaf
   `then` `return`(0);

2. `if` t is a P-node $\{H$ *disconnected*$\}$
   `then` $sum \leftarrow 0$;
   `for` each child t' of t in $T(G)$ `do`
   $sum \leftarrow sum + process(t')$;
   `return`($sum + 1$);

3. `if` t is an S-node $\{\overline{H}$ *disconnected*$\}$
   `then` $sum \leftarrow 0$;
   `for` each child t' of t in $T(G)$ `do`
   $sum \leftarrow sum + process(t')$;
   `return`(sum);

4. `if` t is an N-node $\{H$ *spider*$\}$
   `then` $k \leftarrow$ the size of the partition sets S and K of the spider H;
   $\ell \leftarrow process(t')$, where t' is the node of $T(G)$ which corresponds to the partition set R;
   `return`($k + \ell$);

The correctness of the algorithm follows from Lemma 3.1. We next compute its time and space complexity. Let n and m be the number of vertices and edges of the input graph G. Step 1 of the algorithm can be executed in $O(n + m)$ time and space [9, 10, 17, 25]; we also note that the modular decomposition tree $T(G)$ can be easily obtained from the so-called ps-tree of G [22], which can be computed in $O(n + m)$ time and space using the algorithm in [21] which is simpler than the available modular decomposition algorithms. The subroutine process( ) is called exactly once for each of the $O(n)$ nodes of the tree $T(G)$, and in addition to the time required by any recursive calls, the processing of such a node t takes $O(d_{T(G)}(t) + 1)$ time and $O(1)$ space, where $d_{T(G)}(t)$ is the number of children of node t in $T(G)$. Thus, the total time for the processing of all the nodes of the tree $T(G)$ is $O(n)$; the space needed is $O(n)$. In summary, we have:

Theorem 3.2. *The number of minimal separators of a P_4-sparse graph G on n vertices and m edges can be computed in $O(n + m)$ time using $O(n + m)$ space.*

Algorithm MinSepNum can be easily modified to report all the minimal separators of G based on their characterizations given in Lemma 3.1; in light of Theorem 3.1, the resulting algorithm runs in $O(n + m)$ time and requires $O(n + m)$ space.

4 Finding a Max-Size Subgraph that has a Perfect Matching

In this section we give an optimal sequential algorithm for Problem 2, i.e., the problem of finding a maximum-size subgraph which has a perfect matching. It is important to note that such a subgraph is an induced subgraph; the condition that this subgraph be of maximum size requires that the subgraph be the subgraph induced by its vertex set. Below, we show that any maximum-size subgraph of a graph G, which has a perfect matching, is induced by the matched vertices in a maximum matching of G (Lemma 4.1); then, we prove the property that characterizes a maximum-size subgraph of G, among the subgraphs that are induced by the vertices of maximum matchings of G (Observation 4.1).

Lemma 4.1. *Let G be a graph and let G_1 be a maximum-size subgraph of G which has a perfect matching. Then, the subgraph G_1 is induced by the vertices participating in a maximum matching of G.*

Proof: Let M_{max} be a maximum matching of G, whose vertex set is V_{max}; if $n_{max} = |V_{max}|$ then any matching involving n_{max} matched vertices is a maximum matching of G. If $V_1 \subseteq V(G)$ is the vertex set of the subgraph G_1, then, because G_1 has a perfect matching, $|V_1| \leq n_{max}$. We will show that $|V_1| = n_{max}$, because then any perfect matching of G_1 is a maximum matching of G. It suffices to show that $|V_1| \geq n_{max}$. Suppose, for contradiction, that $|V_1| < n_{max}$. Then, we have that $V_1 \not\subseteq V_{max}$, since otherwise the vertices in V_{max} would induce a subgraph that has a perfect matching and is of size larger than that of G_1, in contradiction to the maximality of G_1. Yet, we will show that we can construct a matching of G whose vertex set has cardinality n_{max} and is a proper superset of V_1; this will yield a contradiction to the maximality of G_1.

Let M_1 be a perfect matching of G_1 (the vertices in V_1 are all matched in the matching M_1) and let us consider the graph H spanned by the non-common edges of the matchings M_1 and M_{max}. Then, the vertices in H have degree at most 2; in particular, the vertices in $V_1 - V_{max}$ have degree exactly 1 and are incident on an edge that participates in M_1. The fact that the degrees of the vertices of H do not exceed 2 implies that each connected component of H is either a path or a cycle. Additionally, edges of M_{max} and M_1 alternate on these paths and cycles.

Let us consider a vertex $x \in V_1 - V_{max}$ and let $\rho = v_0 v_1 \cdots v_t$, $t \geq 1$, be the path (connected component) of H to which x belongs; since x is an endpoint of ρ, we assume without loss of generality that $x = v_0$. Then, the length of ρ cannot be 1; otherwise, $v_1 \in V_1 - V_{max}$, which implies that the edges in M_{max} and the edge xv_1 define a matching of G larger than M_{max}, in contradiction to the optimality of M_{max}. In fact, the length of ρ cannot be odd: if $t = 2q+1$, where $q \geq 1$, then the edges $v_{2i-1}v_{2i}$, $1 \leq i \leq q$, belong to M_{max} and the edges $v_{2i}v_{2i+1}$, $0 \leq i \leq q$, belong to M_1; thus, the vertices $v_1, v_2, \ldots, v_{t-1}$ belong to $V_1 \cap V_{max}$ and the vertices v_0, v_t belong to $V_1 - V_{max}$, which implies that we can replace all the edges $v_{2i-1}v_{2i}$, $1 \leq i \leq q$, by the remaining edges and obtain a matching of G with vertex set $V_{max} \cup \{x, v_t\}$, again in contradiction to the maximality of M_{max}. Therefore, the path ρ has even length, say, $t = 2q$ where $q \geq 1$. Then, as above, the edges $v_{2i-1}v_{2i}$, $1 \leq i \leq q$, belong to M_{max}, the edges $v_{2i}v_{2i+1}$, $0 \leq i \leq q-1$, belong to M_1, the vertices $v_1, v_2, \ldots, v_{t-1}$ belong to $V_1 \cap V_{max}$ and $v_0 \in V_1 - V_{max}$ and $v_t \in V_{max} - V_1$. Thus, if we replace all the edges $v_{2i-1}v_{2i}$, $1 \leq i \leq q$, by the remaining edges, we obtain a matching of G with vertex set $\big(V_{max} \cup \{v_0\}\big) - \{v_t\}$; this matching has n_{max} matched vertices among which we find vertex x whereas the vertex $v_t \in V_{max} - V_1$ is left unmatched.

By doing the above process for each vertex in $V_1 - V_{max}$ and by taking into account that the paths in H to which the vertices in $V_1 - V_{max}$ belong are disjoint, we obtain a matching M of G with vertex set V_M such that $|V_M| = n_{max}$ and $V_1 \subset V_M$. But then, the subgraph $G[V_M]$ has a perfect matching and is of size larger than the size of $G_1 = G[V_1]$, a contradiction to the maximality of G_1. Therefore, the number of vertices of a maximum-size subgraph of G which has a perfect matching is n_{max} and thus is induced by the vertices of a maximum matching of G. ∎

Lemma 4.1 implies that each maximum-size subgraph of a graph G which has a perfect matching is a maximum-size subgraph of G induced by the vertices of a maximum matching of G. Therefore, in the following, we will be referring to the problem of finding a maximum-size subgraph that has a perfect matching while concentrating on maximum-size subgraphs induced by maximum matchings. Then, the following observation allows us to characterize the maximum matching that will yield a maximum-size induced subgraph of G which we seek.

Observation 4.1. *Among all maximum matchings of a graph G, any one whose vertices induce a maximum-size subgraph of G exhibits the minimum sum of degrees of unmatched vertices.*

Proof: Observe that any two vertices left unmatched during the computation of a maximum matching are not adjacent; otherwise, the edge connecting them would produce a larger matching. Hence, the number of edges of the subgraph of G induced by the vertices participating in a maximum matching is equal to the total number of edges of G minus the sum of the degrees of the unmatched vertices. Therefore, in order to obtain a maximum-size subgraph of G induced by the vertices of a maximum matching of G, we need to find a maximum matching of G such that the unmatched vertices have the smallest sum of degrees in G. ∎

Based on Observation 4.1, one might think that, if we know the number k of vertices left unmatched in a maximum matching of G, the maximum-size subgraph of G that we seek can be obtained by removing the k vertices of G of smallest degrees. This is not however the case: consider for example the graph H on 14 vertices $\{v, x_1, x_2, x_3, y_1, \ldots, y_5, z_1, \ldots, z_5\}$ where v is adjacent to all the remaining vertices, $H[\{x_1, x_2, x_3\}]$ is a complete graph on 3 vertices, and $H[\{y_1, \ldots, y_5\}]$ and $H[\{z_1, \ldots, z_5\}]$ are complete graphs on 5 vertices each; H has a maximum matching involving 12 vertices, yet, the subgraph induced by removing any two of the vertices $\{x_1, x_2, x_3\}$ (which exhibit the smallest degrees in H) does not have a perfect matching.

The following lemma establishes another useful property pertaining to a maximum matching.

Lemma 4.2. *Let G be a graph, M_{opt} be a maximum matching of G inducing a subgraph of G whose size is maximum (among all subgraphs induced by the vertices of a maximum matching), M be any other maximum matching of G, and U_{opt} (resp. U) be the set of vertices of G left unmatched by M_{opt} (resp. M). Then there is a bijection $f : U \to U_{opt}$ such that for each vertex $x \in U$, $degree(f(x)) \leq degree(x)$ in G, where by $degree(v)$ we denote the degree of vertex v in G.*

Proof: We work in a fashion similar to that used in the proof of Lemma 4.1. Let E be the set of edges of G which belong to exactly one of the matchings M_{opt} and M, and let A be the set of vertices incident on at least one edge in E. We define the graph H which has vertex set $V(H) = A \cup U_{opt} \cup U$ and edge set E. Then, every vertex of H is of degree ≤ 2 in H, and thus each connected component of H is a single vertex, a path, or a cycle, and edges of M_{opt} and M alternate on these paths and cycles. The latter observation implies that every cycle is of even length. Every path is of even length as well: suppose that there were a path ρ of odd length; then, the edges incident upon both its endpoints would belong to the same matching, say, to M without loss of generality; but then, we would obtain a matching of larger size by substituting in M_{opt} the (fewer in number) even-numbered edges in ρ (if any) by the odd-numbered edges.

For any vertex $x \in U$, x must be of degree ≤ 1 in H since it is unmatched in M. Thus, x either is isolated if it belongs to U_{opt} as well, or is the endpoint of a path (of even length) otherwise. In the former case, we set $f(x) = x$ and thus $f(x) \in U \cap U_{opt}$; in the latter, we set $f(x)$ equal to the other endpoint of the path, which implies that $f(x) \in U_{opt} - U$ since each such path is of even length. The definition of $f(\,)$, the fact that the paths in the graph H are disjoint, and that $|U - U_{opt}| = |U_{opt} - U|$ imply that the function f is a bijection from U to U_{opt}.

Finally, for any vertex $x \in U$, we have $degree(f(x)) \leq degree(x)$ in G: if $degree(f(x)) > degree(x)$, then $f(x) \neq x$ and x would be the endpoint of a path ρ in H; then, by substituting in M_{opt} the odd-numbered edges of ρ starting at x (note that the edge of ρ incident on x belongs to M_{opt}) by the even-numbered edges, we would obtain a maximum matching which leaves unmatched the vertices in $\big(U_{opt} - \{f(x)\}\big) \cup \{x\}$, whose sum of degrees in G is less than that of the unmatched vertices in U_{opt}, in contradiction to the optimality of M_{opt} (see Observation 4.1). ∎

Our algorithm relies on the following lemma as well.

Lemma 4.3. *Let G be a spider with partition sets S, K, and R as described in Section 2.1, and let $G[R]$ be the subgraph of G induced by R. If a maximum-size subgraph of $G[R]$ induced by a maximum matching results after removal of the vertices in $U \subseteq R$, then*

(i) if $|U| \leq |S|$, a maximum-size subgraph of G induced by a maximum matching is $G - X$ where X is an arbitrary subset of S of cardinality $|U|$;

(ii) if $|U| > |S|$, a maximum-size subgraph of G induced by a maximum matching is $G - (S \cup Y)$ where Y is the set of the $|U| - |S|$ vertices of U of smallest degrees in G.

Proof: First, observe that any maximum matching of the graph G involves $|V(G)| - |U|$ vertices. Clearly, it cannot involve fewer vertices as the subgraph $G[R]$ has a matching involving $|R - U|$ vertices and the subgraph $G[S \cup K]$ has a perfect matching (both in the case of a thin and of a thick spider). On the other hand, if there existed a matching involving more vertices, then by ignoring the edges of the matching incident on the vertices in K we get a matching of $G[R]$ (note that a vertex in S can only be matched to a vertex in K); since the number of such edges is at most $|S|$, the resulting matching of $G[R]$ would involve more than $|R - U|$ vertices, in contradiction to the definition of the set U.

Both in cases (i) and (ii) in the statement of the lemma, the proposed subgraphs of G have exactly $|V(G)| - |U|$ vertices, and the vertices left out of the subgraph have the smallest degrees in G; see Observation 2.2. The proof of the lemma will be complete if we show that each of the two subgraphs has a perfect matching, which will then be a maximum matching of G. If $|U| \leq |S|$, then we extend the matching of $G[R]$ by matching the vertices in U to $|U|$ vertices in K and the remaining vertices in K to the vertices in $S - X$; this is a perfect matching of the subgraph $G - X$. If $|U| > |S|$, then we extend the matching of $G[R]$ by matching the vertices in $U - Y$ (which are $|S|$ in number) to the vertices in K; the resulting matching is a perfect matching of the subgraph $G - (S \cup Y)$. ∎

Our algorithm takes advantage of the md-tree (modular decomposition tree) $T(G)$ of the input graph G. To simplify the computations, we "binarize" the S-nodes and P-nodes of the tree $T(G)$ (see [19]). The algorithm processes the resulting tree $T'(G)$ as follows: in each node t of the tree, it computes a maximum matching for the subgraph $G(t)$ of G corresponding to the subtree of $T'(G)$ rooted at t, while at the same time minimizing the sum of the degrees of the vertices of $G(t)$ left unmatched.

Algorithm MaxSubgraph

Input: a P_4-sparse graph G.

Output: a maximum-size subgraph of G which has a perfect matching.

1. Compute the degrees of all the vertices in the graph G and store them in an array;
2. Construct the md-tree $T(G)$ of G;
   Make each S-node or P-node of $T(G)$ binary, obtaining the modified modular decomposition tree $T'(G)$;
3. Execute the subroutine *process*(*root*), where *root* is the root node of the modified md-tree $T'(G)$; the sought subgraph is the subgraph $G - U$, where U is the set of vertices returned by the subroutine.

where the description of the subroutine *process*( ) is as follows:

process(node t)

Input: node t of the modified md-tree $T'(G)$ of the input graph G.

Output: the set U_t of vertices whose removal leaves a maximum-size subgraph of $G(t)$ which has a perfect matching, where $G(t)$ is the subgraph of G corresponding to the subtree of $T'(G)$ rooted at node t.

1. `if` t is a leaf
   `then` `return`($\{v\}$), where v is the vertex associated with the leaf t;
2. `if` t is an N-node
   `then` $S \leftarrow$ partition set S of spider;
   $U_R \leftarrow$ *process*(child of t which is the root of the subtree for $G[R]$);
   `if` $|U_R| \leq |S|$
   `then` $X \leftarrow$ arbitrary subset of S of cardinality $|U_R|$;
   `else` $Y \leftarrow$ set of $|U_R| - |S|$ vertices of U_R of smallest degrees in G;
   $X \leftarrow S \cup Y$;
   `return`(X);
3. $\{$*node t is an S-node or a P-node that has a left and a right child with associated vertex subsets V_ℓ and V_r respectively*$\}$
   $U_\ell \leftarrow$ *process*(left child of t);
   $U_r \leftarrow$ *process*(right child of t);
   suppose without loss of generality that $|U_\ell| \geq |U_r|$, otherwise swap the two children of t and the corresponding sets;
4. `if` t is a P-node
   `then` `return`($U_\ell \cup U_r$);
5. `if` t is an S-node
   `then` `if` $|U_\ell| = |V_r|$
   `then` `return`($\emptyset$);
   `else` `if` $|U_\ell| < |V_r|$
   `then` `if` $|U_\ell| - |U_r|$ is even
   `then` `return`($\emptyset$);
   `else` $v \leftarrow$ vertex in $V_\ell \cup V_r$ of smallest degree in G;
   `return`($\{v\}$);

```
else      {|U_ℓ| > |V_r|}
      X ← subset of the (|U_ℓ| − |V_r|) vertices of U_ℓ of smallest degrees in G;
      return(X);
```

For each node t of the tree $T'(G)$, subroutine process( ) computes a maximum matching for the subgraph $G(t)$ implicitly; it can be easily modified to store and print such a matching. The correctness of Algorithm MaxSubgraph follows from Lemma 4.4.

Lemma 4.4. *When applied on a P_4-sparse graph G, Algorithm MaxSubgraph correctly computes a maximum-size subgraph of G which has a perfect matching.*

Proof: We need only prove the correctness of subroutine process( ). The proof proceeds inductively on the height of the (sub)tree of the modified md-tree $T'(G)$ of the input graph G rooted at the node t that is currently processed by the subroutine. If the (sub)tree is of height 0, then t is a leaf. Hence, it corresponds to a subgraph on 1 vertex; such a graph has no matching and its single vertex is left unmatched, which is what the subroutine returns (Step 1).

For the inductive hypothesis, we assume that the subroutine process( ) correctly handles any (sub)tree of height at most $h \geq 0$; we show that it correctly handles any (sub)tree of height $h+1$. Let t be the root of such a (sub)tree; then, t is an N-node, a binarized S-node, or a binarized P-node. If t is an N-node, then the graph $G(t)$ is a prime spider (Lemma 2.3) and Lemma 4.3 certifies the correctness of the computation in Step 2 taking into account that the ordering of the vertices of $G(t)$ by their degree in $G(t)$ is identical to the ordering based on their degree in G (note that $V(G(t))$ is a module and thus each vertex in $V(G(t))$ is adjacent to the same vertices in $V(G) - V(G(t))$), and that, by the inductive hypothesis, subroutine process( ) has correctly computed a maximum-size subgraph of $G[R]$ which has a perfect matching. Suppose now that t is a P-node or an S-node; in either case, t has two children. If V_ℓ and V_r are the vertex subsets corresponding to the leaves of the left and right child of t respectively, and U_ℓ and U_r are the sets of unmatched vertices returned by subroutine process( ) respectively, then by the inductive hypothesis, the subgraphs $G[V_\ell - U_\ell]$ and $G[V_r - U_r]$ are maximum-size subgraphs of $G[V_\ell]$ and of $G[V_r]$ which have a perfect matching.

If t is a P-node, then there are no edges of G connecting a vertex in V_ℓ to a vertex in V_r; hence, the optimal solution for the subgraph of G corresponding to t, i.e., induced by $V_\ell \cup V_r$, is the union of the optimal solutions for the subgraphs $G[V_\ell]$ and $G[V_r]$, just as the algorithm does (Step 4).

Finally consider that t is an S-node, and assume without loss of generality (as the algorithm assumes) that $|U_\ell| \geq |U_r|$. Then, if $|U_\ell| = |V_r|$, the subgraph $G[V_\ell \cup V_r]$ has a perfect matching: extend the perfect matching of $G[V_\ell - U_\ell]$ with a matching resulting from an arbitrary bijection from U_ℓ to V_r; subroutine process( ) correctly reports that no vertex remains unmatched in this case. An optimal solution is also produced if $|U_\ell| < |V_r|$: a perfect matching is obtained if the total number of vertices is even, otherwise exactly the vertex of U_r with the smallest degree in G (which has the smallest degree in $G(t)$ as well because $V_\ell \cup V_r$ is a module) is left unmatched (the optimality of this solution follows from Observation 4.1), and this can be easily shown to be feasible since $|U_r| \leq |U_\ell| < |V_r|$.

Consider now the case in which $|U_\ell| > |V_r|$. In this case, the algorithm constructs the set X containing the $|U_\ell| - |V_r|$ vertices in U_ℓ of minimum degrees in G, which it returns as the set of vertices left unmatched; this is indeed a feasible solution because a perfect matching in $G[(V_\ell - X) \cup V_r]$ can be constructed from a matching of $G[V_\ell - U_\ell]$ and a matching resulting from an arbitrary bijection from $U_\ell - X$ to V_r. Suppose for contradiction that this is not an optimal solution for the subgraph $G(t)$ of G corresponding to the subtree of $T'(G)$ rooted at the S-node t; let U be the set of unmatched vertices in an optimal solution for $G(t)$. Then, $|U| = |X|$: the optimality of $G(t)-U$ implies that $|U| \leq |X|$, while if $|U| < |X|$ the fact that the subgraph $G(t)-U$ has a perfect matching would imply that a maximum-size subgraph of $G[V_\ell]$ which has a perfect matching would leave a number of unmatched vertices at most $|U| + |V_r| < |U_\ell|$, in contradiction to the optimality of the solution for $G[V_\ell]$ which leaves unmatched the vertices in U_ℓ. The optimality of $G(t) - U$ and the choice of X also imply that

$$\sum_{v \in U} \text{degree}(v) \; < \; \sum_{v \in X} \text{degree}(v) \; \leq \; \sum_{v \in A} \text{degree}(v) \tag{1}$$

for any subset A of U_ℓ of cardinality $|U_\ell| - |V_r|$, where $degree(v)$ denotes the degree of vertex v in G.

Additionally, $U \subseteq V_\ell$; if not, then the fact that U cannot contain both a vertex in V_ℓ and V_r (for otherwise two such vertices would be matched resulting in a larger matching) would imply that $U \subseteq V_r$, which would in turn imply that $G[V_\ell]$ has a matching leaving unmatched at most $|V_r| - |U| \leq |V_r| < |U_\ell|$ vertices, a contradiction. We note however that it does not have to be the case that $U \subseteq U_\ell$; nevertheless, we will show that there exists a bijection $g : U \to W$, where $W \subseteq U_\ell$, such that for each vertex $x \in U$, $degree(g(x)) \leq degree(x)$.

Let us consider the restriction of the matching which yields the optimal solution for the graph $G(t)$ (and leaves unmatched the vertices in U) to the vertex set V_ℓ; if Z is the set of vertices left unmatched by the resulting matching M', then clearly $U \subseteq Z$ and $|Z| \leq |U| + |V_r| = |U_\ell|$. Thus, since the optimal matching for this subgraph (by the inductive hypothesis) leaves unmatched the vertices in U_ℓ, $|Z| = |U_\ell|$ and the matching M' is maximum for the subgraph $G[V_\ell]$. Then, Lemma 4.2 applied on the two matchings for the graph $G[V_\ell]$, which leave unmatched the vertices in Z and U_ℓ respectively, yields that there exists a bijection $f : Z \to U_\ell$ such that for each $x \in Z$, $degree(f(x)) \leq degree(x)$. Then, the desired bijection g is the restriction of the bijection f to the domain U, and the set W is the set of images $f(x)$ of the vertices x in U. The properties of the bijection g imply that

$$\sum_{v \in W} \text{degree}(v) \; \leq \; \sum_{v \in U} \text{degree}(v),$$

which comes into contradiction with Inequality (1), since W is a subset of U_ℓ of cardinality $|U| = |U_\ell| - |V_r|$. Therefore, the solution produced by the algorithm for the case when $|U_\ell| > |V_r|$ is also optimal. ∎

Time and Space Complexity of Algorithm MaxSubgraph: Let n and m be the number of vertices and edges of the input P_4-sparse graph G. Then, the degrees of the vertices of G can be computed in $O(n+m)$ time and can be stored in $O(n)$ space, so that Step 1 of the algorithm takes $O(n+m)$ time and $O(n)$ space. The md-tree $T(G)$ can be constructed in $O(n+m)$ time and space [9, 25, 10, 17] and has $O(n)$ size; we also note that the tree $T(G)$ can be easily obtained from the ps-tree of G [21], which can be computed in $O(n + m)$ time and space using the algorithm in [21] which is simpler than the available modular decomposition algorithms. Binarizing the S- and P-nodes takes $O(n)$ time and the resulting tree $T'(G)$ has $O(n)$ size. Thus, Step 2 takes $O(n + m)$ time and space as well.

Let us now bound the time and space needed by all the executions of subroutine process( ). The set of unmatched vertices returned by a call to process( ) is stored in an unordered linked list so that linking two such lists takes constant time. Observe that each node t of the tree $T'(G)$ is processed exactly once. If t is a leaf, then the processing of t takes constant time (Step 1). If t is an N-node, then t's processing takes $O(|S| + |U_R|)$ time (see Step 2): the set S is readily available from the md-tree; forming the set Y requires copying the vertices in U_R along with their degrees in an array, applying the linear-time selection algorithm [6] on this array to locate the value of the $(|U_R| - |S|)$-th smallest degree, and using this value to partition the vertices in U_R based on their degrees (special attention is needed for the vertices with degree equal to the partitioning value). In turn, the processing of a P-node takes constant time (thanks to the linked list representation of the set of unmatched vertices), while the processing of an S-node takes $O(|V_\ell| + |V_r|)$ time (again, forming the set X requires the application of the linear-time selection algorithm). We get a bound on the time taken for the processing of the entire tree $T'(G)$ by using the following crediting scheme: we credit each leaf and each (binary) P-node with 1 credit; we credit each N-node (corresponding to a spider with partition sets S, K, and R) with the number of edges of G connecting vertices in S and K, and vertices in K and R, which is at least equal to $|S| + |K| \cdot |R|$; we credit each (binary) S-node with 1 plus the number of edges of G connecting a vertex associated with a leaf in the left subtree of the S-node to a vertex associated with a leaf in the right subtree, that is, with $1 + |V_\ell| \cdot |V_r|$ credits. Since $|U_R| \leq |R|$, $|K| \geq 2$, and $|V_\ell| + |V_r| \leq 1 + |V_\ell| \cdot |V_r|$, the time taken is bounded by a constant multiple of the number of credits. Then, because each edge of G contributes at most one credit in our crediting scheme and because the size of $T'(G)$ is $O(n)$, the time required for the completion of Step 3 of Algorithm MaxSubgraph is $O(n + m)$. Clearly, the space needed by subroutine process( ) is $O(n)$.

The results of this section can be summarized in the following theorem.

Theorem 4.1. *Let G be a P_4-sparse graph on n vertices and m edges. The problem of finding a maximum-size subgraph of G admitting a perfect matching is solved in $O(n+m)$ time and space.*

5 Concluding Remarks

In this paper, we determined the minimal separators of the P_4-sparse graphs and showed that the number of minimal separators of such a graph (or cograph by inclusion) on n vertices and m edges, does not exceed $2n/3$ and that their description size is at most $4m/3$. This result enables us to include these two classes of graphs to the collection of graph classes, such as chordal graphs and weakly chordal graphs, which have a linear number of minimal separators. We also presented a linear-time algorithm for computing the number of minimal separators of a P_4-sparse graph, which can be easily modified to report the minimal separators within the same time and space complexity.

The obvious open problem is to compute the number of minimal separators in other classes of graphs; as a first step, one could try to work on classes that properly contain the P_4-sparse graphs, e.g., the classes of the P_4-laden, P_4-lite, P_4-extendible, and P_4-tidy graphs (see [4] for an exposition of such classes).

We have also characterized the maximum-size subgraphs of a P_4-sparse graph which admit a perfect matching, and presented a linear-time algorithm for the problem of finding such a subgraph; these results also hold for the class of cographs.

Motivated by this work, it would be interesting to consider the related problem where edges are added so that the resulting graph has a perfect matching while remaining in the same class of graphs, that is,

- Given a P_4-sparse graph (or cograph) G, find the minimum number of edges which need to be added to the edge set of G such that the resulting graph is a P_4-sparse graph (or cograph) and admits a perfect matching.

We expect that the structural results and the algorithmic approach used in this paper can help develop efficient algorithms for solving this problem as well.

References

[1] A. Berry, Graph extremities and minimal separation, *Proc. 4th Int. Conference JIM 2003: Knowledge Discovery and Discrete Mathematics*, Metz, France, 2003.

[2] A. Berry and J.-P. Bordat, Triangulated and weakly triangulated graphs: Simpliciality in vertices and edges, *Proc. 6th Int. Conference on Graph Theory*, Luminy, France, 2000.

[3] A. Berry, J.-P. Bordat, and O. Cogis, Generating all the minimal separators of a graph, *Int. J. Foundations of Comput. Sci.* **11** (2000) 397–404.

[4] A. Brandstädt, V.B. Le, and J.P. Spinrad, *Graph Classes: A Survey*, SIAM Monographs on Discrete Mathematics and Applications, 1999.

[5] A. Bretscher, D. Corneil, M. Habib, and C. Paul, A simple linear time LexBFS cograph recognition algorithm, *Proc. 29th Int. Workshop on Graph-Theoretic Concepts in Comput. Sci. (WG'03)*, LNCS 2880 (2003) 119–130.

[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms* (2nd edition), MIT Press, Inc., 2001.

[7] D.G. Corneil, S. Olariu, and L. Stewart, Asteroidal triple-free graphs, *SIAM J. Discrete Math.* **10** (1997) 399–430.

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

[9] A. Cournier and M. Habib, A new linear algorithm for modular decomposition, *Proc. 19th Int. Colloquium on Trees in Algebra and Programming (CAAP'94)*, LNCS 787 (1994) 68–84.

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

[11] E. Dahlhaus, P. Hammer, F. Maffray, and S. Olariu, On domination elimination orderings and domination graphs, *Proc. 20th Int. Workshop on Graph-Theoretic Concepts in Comput. Sci. (WG'94)*, LNCS 903 (1994) 81–92.

[12] J.L. Fouquet, I. Parfenoff, and H. Thuillier, An O(n) time algorithm for maximum matchings in P_4-tidy graphs, *Inform. Process. Lett.* **62** (1999) 281–287.

[13] D.R. Fulkerson and O.A. Gross, Incident matrices and interval graphs, *Pacific J. Math.* **15** (1965) 835–855.

[14] V. Giakoumakis and J. Vanherpe, On extended P_4-reducible and extended P_4-sparse graphs, *Theoret. Comput. Sci.* **180** (1997) 269–286.

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

[16] M.C. Golumbic and C.F. Goss, Perfect elimination and chordal bipartite graphs, *J. Graph Theory* **2** (1978) 155–163.

[17] M. Habib, F. de Montgolfier, and C. Paul, A simple linear-time modular decomposition algorithm for graphs, using order extension, *Proc. 9th Scandinavian Workshop on Algorithm Theory (SWAT'04)*, LNCS 3111 (2004) 187–198.

[18] C. Hoàng, Perfect graphs, *Ph.D. Thesis*, McGill University, Montreal, Canada, 1985.

[19] J. JáJá, *An Introduction to Parallel Algorithms*, Addison-Wesley, 1992.

[20] B. Jamison and S. Olariu, P_4-reducible graphs a class of tree representable graphs, *Studies Appl. Math.* **81** (1989) 79–87.

[21] B. Jamison and S. Olariu, Recognizing P_4-sparse graphs in linear time, *SIAM J. Comput.* **21** (1992) 381–406.

[22] B. Jamison and S. Olariu, A tree representation for P_4-sparse graphs, *Discrete Appl. Math.* **35** (1992) 115–129.

[23] B. Jamison and S. Olariu, Linear-time optimization algorithms for P_4-sparse graphs, *Discrete Appl. Math.* **61** (1995) 155–175.

[24] H. Lerchs, On cliques and kernels, *Technical Report*, Department of Computer Science, University of Toronto, March 1971.

[25] R.M. McConnell and J. Spinrad, Linear-time modular decomposition and efficient transitive orientation of comparability graphs, *Proc. 5th ACM-SIAM Symp. on Discrete Algorithms (SODA'94)* (1994) 536–545.

[26] S. Micali and V.V. Vazirani, An $O(\sqrt{n}\,m)$ algorithm for finding maximum matching in general graphs, *Proc. 21st Annual IEEE Symposium on Foundation of Computer Science (FOCS'80)*, (1980) 17–27.

[27] F.S. Roberts, Indifference graphs, *Proof Techniques in Graph Theory* (F. Harary ed.), Academic Press (1969) 139–146.

[28] D.J. Rose, Triangulated graphs and the elimination process, *J. Math. Anal. Appl.* **32** (1970) 597–609.