David B. Chandler, Maw-Shang Chang,
Ton Kloks, Jiping Liu, Sheng-Lung Peng

Probe Graph Classes

October 17, 2012

Contents

Part I

Introductory Matter

1

Introduction

UNFORTUNATELY, 'most' graph algorithmic problems that appear in practical situations are NP-complete for graphs in general. In practical situations however, the graphs at hand are rarely general: some underlying structure of the graph class of interest often allows an efficient algorithm that solves the problem. Therefore, much effort is put into research to solve NP-complete problems for graphs restricted to certain graph classes. Certain problems that turn up in the Human Genome Project can be expressed as problems on interval graphs. To analyze a long strand of DNA, enzymes are used to cut the DNA sequence into smaller fragments called clones. The clones are reproduced many times for further research. To reconstruct the DNA strand, tests are performed to determine whether a pair of clones overlap in the longer DNA strand. If these tests were run on every pair of clones, the problem would map nicely to the interval graph recognition problem. To reduce the number of physical experiments that are needed, one selects a subset of the clones, called probes, and tests for overlaps between two clones if and only if at least one of the clones is a probe. This gives rise to a new graph theoretic model. Probe interval graphs were introduced in 1994 by Zhang [192] and used in [193, 194] to model certain problems in physical mapping of DNA. Clones correspond to vertices of a graph, where vertices are labeled as either probes or nonprobes. The objective is to find a mapping of the vertices to intervals on the line such that two vertices are adjacent if and only if the intervals have a nonempty intersection and at least one of the vertices is a probe. In other words; the input to the problem is a graph G and a subset of probe vertices. The other vertices, the nonprobes, form an independent set in G. The objective is to add edges between certain nonprobe vertices such that the graph becomes an interval graph [92, Chapter 4]. Generalizing this concept, we introduce the following definition of probe graphs of graph classes:

Definition 1.1. *Let $\mathcal{G}$ be a class of graphs. A graph $G = (V, E)$ is a* probe graph *of $\mathcal{G}$ if its vertex set can be partitioned into a set of* probes $\mathbb{P}$ *and an* independent set *of* nonprobes $\mathbb{N}$*, such that G can be embedded in a graph of $\mathcal{G}$ by adding edges between certain nonprobes.*

If the partition of the vertices of a graph G into a set of probes $\mathbb{P}$ and a set of nonprobes $\mathbb{N}$ is part of the input we call G a *partitioned probe graph of* $\mathcal{G}$ if G can be embedded into a graph of $\mathcal{G}$ by adding edges between certain vertices of $\mathbb{N}$. We denote a partitioned graph as $G = (\mathbb{P} + \mathbb{N}, E)$, and when this notation is used it is to be understood that $\mathbb{N}$ is an independent set. We will refer to the class of (partitioned) probe graphs of the class of `(XXX)` graphs as (partitioned) probe `(XXX)` graphs where `(XXX)` is the name of a graph class.

To gain insight into the graph class of probe interval graphs, attention was drawn to probe chordal graphs in [13, 89]. A year later, in 2004, the recognition of partitioned and unpartitioned probe chordal graphs was handled in [14]. The algorithm for the partitioned case runs in $O(nm)$ time, while the unpartitioned case takes $O(n^2m)$ time. Between whiles, efficient algorithms for the recognition of partitioned probe interval graphs turned up in [129, 147]. The first of these algorithms runs in $O(n^2)$ time, while the second one runs in $O(n + m \log n)$ time. Finally, a recognition algorithm for unpartitioned probe interval graphs appeared in [36]. According to [14], probe chordal graphs also have immediate applications in certain reconstruction problems of phylogenies. At present, the study of most other probe graph classes is of (great) interest mainly for theoretical reasons. The study of a probe graph class first of all establishes demarcations on the *robustness* of the graph class with respect to irresolute inputs. Partitioned probe interval graphs were introduced for this purpose. It also brings to light many interesting, sometimes unforeseen properties of the new graph class in question. We hope that this book convinces the reader that the study of probe graphs of graph classes is highly illuminating and satisfying. We aim at gaining interest by showing many nice structural results of these new graph classes and by illustrating a wide variety of techniques applicable for the recognition of them.

The recognition problem of partitioned probe graph classes can be seen as a subcase of the graph sandwich problem [82, 88, 91]. In the general sandwich problem for a graph property π, one is given two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $E_1 \subseteq E_2$ and the question is whether there exists a graph $G = (V, E)$ such that $E_1 \subseteq E \subseteq E_2$ and which satisfies property π. In [88] it is shown that the problem can be solved in polynomial time when π is the property of being a threshold graph, a splitgraph, or a cograph. The problem is shown to be NP-complete for comparability graphs, permutation graphs, and several other graph classes. Given a partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$, we

can define the graph $G_1 = G$ and G_2 as the graph obtained from G by adding all edges between nonprobes. The question whether G is a partitioned probe graph of a graph class $\mathcal{G}$ thus translates directly to the graph sandwich problem with π as the property of being a graph of $\mathcal{G}$. In this book we show that the recognition problem for partitioned probe graphs is solvable in polynomial time for many more graph classes than the general sandwich problem.

2

Preliminaries

A *graph* G is a pair $G = (V, E)$, where $V \neq \varnothing$ is a finite set, the elements of which are called the *vertices* of G and where E is a set of two-element subsets of V, called the *edges*. For our convenience we denote an edge e as $e = (x, y)$, rather than $e = \{x, y\}$ or $e = xy$, and we say that x and y are *adjacent* in G. Two vertices which are adjacent are called *neighbors* of each other. For a vertex x we denote its set of neighbors by $N(x)$ and call this set the *neighborhood* of x. We let $N[x] = N(x) \cup \{x\}$ and call this set the *closed neighborhood* of x. If x and y are adjacent we say that they are the *endvertices* of the edge (x, y). To indicate that two vertices x and y are adjacent we sometimes write $x \sim y$. We write $n = |V|$ for the number of vertices and $m = |E|$ for the number of edges.

For two sets A and B we write $A + B$ and $A - B$ instead of $A + B$ and $A \setminus B$ respectively. We write $A \subseteq B$ if A is a subset of B with possible equality and we write $A \subset B$ if A is a subset of B and $A \neq B$. For a set A and an element x we write $A + x$ instead of $A + \{x\}$ and $A - x$ instead of $A - \{x\}$.

For a graph $G = (V, E)$ and a subset $S \subseteq V$ we write $G[S]$ for the subgraph of G *induced* by S, *i.e.*, the subgraph with vertex set S and edges those elements of E with both endvertices in S. For a subset $W \subseteq V$ we write $G - W$ for the subgraph induced by $V - W$. For a vertex x we write $G - x$ rather than $G - \{x\}$. For a subset $W \subseteq V$ of vertices we write $N(W) = \cup_{x \in V} N(x) - W$.

The complement of a graph G, denoted as $\overline{G}$, is the graph in which two vertices are adjacent exactly when they are not adjacent in G. A *path* in a graph $G = (V, E)$ is a sequence of vertices $[x_1, \ldots, x_k]$ where $(x_i, x_{i+1}) \in E$ for $i = 1, \ldots, k-1$. The path is *chordless* if there are no other edges in the induced subgraph $G[\{x_1, \ldots, x_k\}]$ than the ones mentioned above. A graph isomorphic to a chordless path with k vertices is denoted as P_k. A cycle in a graph G is a sequence of vertices $[x_1, \ldots, x_k]$ where $\{x_i, x_{i+1}\}$ for $1 = 1, \ldots, k - 1$ and $\{x_1, x_k\}$ are edges in G. The cycle is *chordless* if there are no other edges

than the ones mentioned above in $G[\{x_1, \ldots, x_k\}]$. We denote a graph which is isomorphic to a chordless cycle with k vertices by C_k.

A *separator* (or cutset) in a graph $G = (V, E)$ is a subset $S \subset V$ of vertices such that the vertices of $G - S$ can be partitioned into two non–empty sets V_1 and V_2 such that there is no edge with one end–vertex in V_1 and the other in V_2. A graph G is *disconnected* if the empty set is a separator in G. The graph is *connected* otherwise. If a graph G is disconnected then the *components* of G are the maximal subsets of vertices that induce connected subgraphs in G. If S is a separator and C is a component of $G - S$ then C is a *full component* of S if every vertex of S has at least one neighbor in C. A separator S is *minimal* if it has at least two full components. For any pair of vertices x and y in different full components of a minimal separator S, the separator is called a *minimal* x, y*-separator*.

A *clique* in a graph G is a subset S of its vertices such that every pair of vertices in S is adjacent. A graph isomorphic to a clique with ℓ vertices is denoted by K_ℓ. The *clique number* $\omega(G)$ of a graph G is the maximum number of vertices in a clique of G. The *chromatic number* $\chi(G)$ of a graph G is the minimum number of colors needed to color the vertices such that the end–vertices of every edge receive different colors. It is easy to see that for every graph $\chi(G) \geq \omega(G)$, since, when coloring the vertices of a graph such that adjacent vertices receive different colors, all vertices of any clique in G must receive different colors.

The computation of both numbers $\chi(G)$ and $\omega(G)$ are NP–complete, *i.e.*, it is widely believed that they are not efficiently computable. However, whenever for a graph G equality of these numbers occurs, then there is a polynomial time algorithm to compute it [141]. This is the case because there exists a polynomial time computable value $\theta(G)$ such that $\omega(G) \leq \theta(G) \leq \chi(G)$ for every graph G. The value of $\theta(G)$ can be obtained in polynomial time by linear programming methods [10], pp. 325–356.

A *hole* in a graph G is a chordless cycle of length at least 5. An *antihole* is the complement of such a cycle. A hole or antihole is *odd* if the number of its vertices is odd, otherwise it is even. Consider coloring the vertices of a chordless cycle such that adjacent vertices receive different colors. When the cycle is even, then this can be done by using only two colors, since the two colors can be assigned alternately to the vertices along the cycle. Hence for every *even* cycle the chromatic number is equal to its clique number, *i.e.* 2. However, when the cycle is odd and has at least 5 vertices, one more color is needed, that is $\omega = 2$ and $\chi = 3$ for these graphs. The same phenomenon occurs when coloring the *complements* of these cycles. If the cycle has an odd number of vertices at least 5 then one more color is needed to color the complement of the cycle than the lowerbound as indicated by the clique

number suggests. If the cycle has an even number of vertices, then the clique number and chromatic number of the complement are equal.

A graph G is called *perfect* if for *every induced subgraph* the chromatic number equals its clique number. As shown above, perfect graphs cannot have odd holes or antiholes. Many special classes of perfect graphs have been thoroughly investigated and documented over the last years [26, 86], and many of them find applications in various fields. This is so because for a lot of these classes also many other NP-complete problems are solvable using efficient algorithms, and this solves many problems that occur in practice.

The *perfect graph conjecture*, proved by Lovász in 1972 [141],[1] states that the class of perfect graphs is self-complementary, *i.e.*, a graph G is perfect if and only if its complement $\overline{G}$ is perfect. The *strong* perfect graph conjecture guesses that a graph is perfect if and only if the graph contains no odd hole or antihole. A proof of this conjecture was recently levied in [43].

For a nice appetizer on perfect graphs we refer to [94]. Both conjectures on perfect graphs, the first one saying that the class is self complementary, and the second one saying that they are exactly the graphs without holes or antiholes, are attributed to Berge. The problems find their origin in Shannon's classical paper [175], and Berge says that his attention was drawn to this paper some time in June 1957. With his student Alain Ghouila-Houri, Berge formulated his two most famous conjectures in 1960. Since then both conjectures have made a tremendous impact on graph-theoretical research and the work on algorithmic graph theory and combinatorics. In [16], quoting many famous mathematicians like Rota and Chvátal, it is mentioned that Berge's work helped rescue graph theory and combinatorics from "the slum of mathematics." We leave it in the middle whether this was still necessary at that time. It seems funny, that Berge writes in [9] that Shannon's paper [175] "could have been missed by mathematicians working in algebra and combinatorics." Nowadays, this is almost impossible to imagine! Shannon had a very clearcut motivation for studying and posing part of the questions [175] that were glamorized by Berge into his two most famous conjectures. For a more accurate report on the history we refer to [9], and for an exhibition how it links combinatorics, graph theory, combinatorial optimization, semi–definite programming, polyhedral and convexity theory, and "even" information theory, we refer to the beautiful "favorite theorem" [94] paper of Grötschel.

[1] When informed by Berge, via a postcard, of Lovász' success, Fulkerson, who had worked on the problem for more than a decade, independently finished his own proof within a few hours [16]. Since then, the proof of this theorem has developed into a one–hour presentable classroom appetizer taking up less space than half a page [78].

As already mentioned above, one of the merits of perfect graphs is that the "basic" NP-complete problems CLIQUE, INDEPENDENT SET[2], CHROMATIC NUMBER, and CLIQUE COVER[3] become solvable in polynomial time when restricted to perfect graphs [10]. For other problems that are NP-complete for graphs in general much less is known when the graphs are restricted to the class of perfect graphs, but for many important sub–families also many other problems become efficiently tractable. Recent developments indicate that a tremendous expansion of this knowledge can be expected in the near future due to the discovery of a decomposition for perfect graphs, discussed in some detail below.

Until very recently it was unknown whether deciding if a graph is perfect or not is in NP. Recently a polynomial time recognition algorithm for perfect graphs was announced [54]. The complexity of finding an odd hole or antihole was unknown for quite some time, and until now, no easy algorithm is available. The currently fastest algorithm to detect the presence of odd–length holes or antiholes takes $O(n^{10})$ time, where n is the number of vertices of the graph [54, 164].[4] The proof of the strong perfect graph theorem shows that these graphs go along with an almost elementary decomposition tree with only 4 types of prime graphs and only a few types of internal nodes, *e.g.*;

[2] An *independent set* in a graph G is a clique in the complement $\overline{G}$.

[3] A *clique cover* in a graph G is a coloring of the complement $\overline{G}$.

[4] The algorithm of [54] can also be used to produce an odd hole *or* antihole. In [54] it is mentioned that checking whether a general graph contains an odd hole is still an open problem. At the moment we are not aware of any algorithm that produces an odd hole (if it exists). An odd–hole–free graph recognition algorithm for graphs of bounded clique size has been announced [45]. For *even holes* the present situtation is as follows. In [42] an algorithm appeared that *produces* an even hole if there exists one. However, it should be mentioned that in this paper holes are defined as chordless cycles of length at least 3. Hence the algorithm could produce a "hole" of length 4, and it is not obvious how to augment this algorithm for finding even holes of length more than 5. The algorithm of [42] runs in $O(n^{31})$ time. In this paper it is mentioned that the problem of finding the *shortest* even hole is still open. The situation where the parity constraint is dropped is as follows: A nice characterization of graphs without holes or antiholes appeared in [11]. The algorithm based on this characterization runs in $O(n + m^2)$ time and uses $O(n + m^2)$ space. However, it does not produce a hole or antihole in all cases. The algorithm of [105] produces a hole *or* antihole if there exists one. This algorithm can be implemented to run on $O(n+m^2)$ time and uses $O(n+m)$ space. The algorithm of [164] checks if a graph has a hole and produces one if this is the case. This algorithm can be implemented to run in $(n + m^2)$ time and $O(nm)$ space. The same result is shown for antiholes, *i.e.*, an antihole can be detected and produced in $O(n + m^2)$ time $O(nm)$ space. If the graph does not have a C_5 the space complexity can be reduced to $O(n + m)$. Notice that Bienstock showed that it is NP-hard to find odd holes containing a given vertex.

every *odd-hole-free graph* is either basic or has a double star cutset or a 2-join [54].

Graphs that allow a decomposition tree with a restricted set of prime graphs that can occur as leaves of the tree, and a restricted set of internal nodes which serve as graph decomposition operators, often permit efficient algorithms for certain NP-complete problems. Examples are graphs of bounded treewidth, branchwidth, cliquewidth, &tc, and many other graph classes like planar graphs and cographs which found their origin in alternative characterizations first. Research takes place in two directions. Usually, either one starts with the graph class and tries to find the suitable decomposition tree, or one starts with a certain decomposition tree and tries to analyze the graphs which permit this decomposition. Here one takes a certain decomposition tree and tries to "fit" the graph into this "haute couture." For some decomposition trees, like tree decompositions, the prime graphs consist of single vertex graphs. As an oposite example in this respect one can consider the modular–decomposition tree, where the prime graphs can take almost any shape and size and the decomposition tree is determined solely by the description of the interplay of the internal nodes as modules. Although one type of research usually precedes the other, both directions are researched eventually.

In order to prove the perfect graph theorem researchers guessed for a long time at the suitable decomposition tree.[5] This research led to various alternative decomposition trees, *e.g.*, different types of homogeneous decomposition trees and decomposition trees using star cutsets, or clique cutsets, that appear to be useful in practice to solve certain NP–complete problems. Eventually, two separate groups of researchers found two slightly different decomposition trees (above, we only mentioned the one that appeared in [54]) and proved the correctness of the strong perfect graph theorem along the way. In case a graph is perfect, there exists a decomposition tree where the "basic graphs", or "prime graphs", that occur as leaves of this decomposition tree, are bipartite graphs,[6] linegraphs[7] of bipartite graphs, or the complements of these classes. Notice that, for example, the bipartite graphs are perfect, since they do not contain any odd cycle. It is fairly easy to see that also the other

[5] Quite early, the right collection of prime graphs was conceived.

[6] A graph is *bipartite* if its chromatic number is at most two, *i.e.*, when there exists a partition of the vertices into two sets V_1 and V_2 such that every edge connects a vertex of V_1 with a vertex of V_2. Alternatively, bipartite graphs can be characterized as those graphs without any odd length cycle.

[7] The *linegraph* $L(G)$ of a graph $G = (V, E)$ is the intersection graph of the elements of E, *i.e.*, the vertices of $L(G)$ are the edges of G and two vertices in $L(G)$ are adjacent whenever the corresponding edges in G have a vertex in common.

basic graphs are perfect. For the basic (or: "prime") perfect graphs many NP–complete problems are solvable in polynomial (and usually even linear) time.

A *double star cutset* is a separator S containing two adjacent vertices x and y such that all other vertices of S are adjacent to at least one of these two. It is not difficult to find a double star cutset in a graph in polynomial time: Simply try all possible adjacent vertices x and y and a pair of vertices a and b such that a and b are not adjacent. Then let S be the set of vertices that contains x, y, and all neighbors of x and y except a and b. Finally check if this set S *separates* a and b into different components. In the recognition algorithm for perfect graphs the problem is to find those double star cutsets that are suitable for the decomposition tree, *i.e.*, it is required that they decompose a graph in such a way that the *odd-hole-free property* is maintained. Until now, this is the bottleneck in the recognition algorithm: Double star cutsets that guarantee the preservation of odd-hole-freeness can be found only *after* the graph is first *cleaned*.

A *2-join* is a partition of the vertices into two sets V_1 and V_2 such that each contains nonempty vertex sets A_i and B_i, $i = 1, 2$, such that every vertex of A_1 is adjacent to every vertex of A_2, every vertex of B_1 is adjacent to every vertex of B_2, and there are no further adjacencies between V_1 and V_2. A 2-joins in a graph can also be detected in polynomial time [53]. For the recognition algorithm of perfect graphs they present less difficulties than the double star cutsets: Every 2-join can be used to decompose a graph such that the odd-hole-free property is preserved.

In order to prove the strong perfect graph theorem, two separate groups of researchers joined forces and proved that, after cleaning the graph, it captures the class of perfect graphs in the sense that a decomposition can be obtained from double star cutsets and 2-joins that preserves the odd-hole-free property. The decomposition tree with prime graphs as the leaves and double star cutsets and 2-joins as internal nodes, seems at the moment to be the winning strategy for the recognition algorithm of perfect graphs (after the clean-up before a double star cutset is used), *i.e.*, deciding whether a graph has an odd hole or antihole.

The decomposition tree could be of great importance to find polynomial time combinatorial algorithms for certain NP-complete problems on perfect graphs and other graph classes. Therefore, finding more efficient algorithms to find this decomposition tree and the use of it to solve NP-complete problems is of great interest. For example, it would be of great interest to find *combinatorial* algorithms for problems such as CLIQUE, CHROMATIC NUMBER, and others for the class of perfect graphs. Although more widely applicable, the available algorithms for computing the clique and chromatic number at the moment use linear programming and are of little use in practice. More-

over, the algorithms do not produce a proof that the computed number is the correct one unless the graph is *known* to be perfect. The newly obtained decomposition tree for perfect graphs indicates that more efficient algorithms are within reach for the class of perfect graphs.

Definition 2.1. *Let $\mathcal{G}$ be a class of graphs. A graph G is a* probe graph of $\mathcal{G}$ *if its vertices can be partitioned into a set $\mathbb{P}$ and an independent set $\mathbb{N}$ such that G can be embedded into a graph $G' \in \mathcal{G}$ by adding edges between certain vertices of $\mathbb{N}$. The vertices of $\mathbb{P}$ are called* probes *and those of $\mathbb{N}$ are called* nonprobes*.*

If the partition of the vertices into probes and nonprobes is part of the input, then we call the graph a *partitioned probe graph of* $\mathcal{G}$. In this book we denote a partitioned probe graph of $\mathcal{G}$ as $G = (\mathbb{P} + \mathbb{N}, E)$, and when this notation is used it is to be understood that $\mathbb{N}$ is an independent set in G. If the graph class $\mathcal{G}$ is not specified we call $G = (\mathbb{P} + \mathbb{N}, E)$ a *partitioned graph* if $\mathbb{N}$ is an independent set. We will refer to the class of (partitioned) probe graphs of the class of `(XXX)` graphs as (partitioned) probe `(XXX)` graphs where `(XXX)` is the name of a graph class. We call a graph $G' \in \mathcal{G}$ obtained from G by adding some edges between vertices of $\mathbb{N}$ an *embedding* of G.

Definition 2.2. *Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned graph. Let G^* be obtained from $\overline{G}$ by removing all edges between vertices of $\mathbb{N}$. G^* is called the* sandwich conjugate of G.

One of the main motivations for studying probe graphs of classes of perfect graph is the following observation:

Theorem 2.3 (Probe Sandwich Theorem). *Let $\mathcal{G}$ be a class of graphs which is* self-complementary*, i.e., $H \in \mathcal{G} \Leftrightarrow \overline{H} \in \mathcal{G}$. Let $G = (\mathbb{P}+\mathbb{N}, E)$ be a graph with a partition of its vertices into $\mathbb{P}$ and an independent set $\mathbb{N}$. Then G is a partitioned probe graph of $\mathcal{G}$ if and only if its sandwich conjugate G^* is in the same category.*

Corollary 2.4. *A graph $G = (\mathbb{P} + \mathbb{N}, E)$ is a partitioned probe perfect graph if and only if its sandwich conjugate is likewise.*

Probe graphs of classes of perfect graphs constitute new classes, and it is interesting to see that many probe classes of classes of perfect graphs are still perfect.

Definition 2.5. *A* Meyniel graph *is a graph in which every cycle of odd length at least 5 has at least two chords.*

Meyniel graphs are the graphs without an odd hole nor a cap.[8]

[8] A *cap* is a cycle of length at least 5 with exactly one chord between two vertices at distance 2 in the cycle.

Theorem 2.6. *The only graph classes for which all probe graphs are perfect, are classes of Meyniel graphs.*

Proof. The probe Meyniel graphs are known in the literature as the *slim* graphs. The perfectness of slim graphs was shown in [110]. For alternative proofs see [43, 111]. If a graph has an odd hole then obviously it cannot be perfect. If a graph does not have an odd hole but, instead, a cap, we can take the two endvertices of the chord as nonprobes and the other vertices as probes. Thus we obtain a probe graph with an odd hole. □

Meyniel graphs can be recognized in polynomial time [31]. As far as we know, the recognition of slim graphs is still an open problem. We conjecture that at least the partitioned case is recognizable in polynomial time:

Conjecture 2.7. There exists a polynomial time recognition algorithm for partitioned probe Meyniel graphs.

In the following theorem we show a similar characterization for the graph classes for which all probe graphs are weakly chordal. For the house, domino, and gem we refer to Figure 8.1 on page 89. A *sun* is a graph obtained from an even cycle of length at least 6 in which edges are added to make a maximum independent set into a clique. For a 3-sun we refer, *e.g.*, to Figure 3.1 on page 22. If some edges of the clique are possibly missing, but the graph is still chordal, it is called a *trampoline*. Farber and Chang [71, Lemma 4.5] observed that every trampoline has a sun as an induced subgraph. In the following, a *partial* k*-sun* is a graph with $2k$ vertices, of which k vertices induce the *kernel*, which is a Hamiltonian graph, and k other vertices form an independent set. Each vertex of the kernel is adjacent to exactly two vertices of the independent set, and each vertex of the independent set is adjacent to two consecutive vertices of the Hamiltonian cycle of the kernel. Note that we do not require here that G be chordal.

Theorem 2.8. *The probe graphs obtained from a graph are all weakly chordal if and only if the graph contains no induced house, hole, domino, nor sun.*

Proof. A graph whose probe graphs are all weakly chordal clearly cannot contain a house, a hole, a domino, or a sun. In each case we get a hole in the probe graph by deleting the edges joining a certain set of nonprobes. We show by contradiction that the probe graphs of the graph class with none of these induced subgraphs are weakly chordal.

Let G be a graph containing a hole or an antihole and suppose H is some embedding of G which contains no induced house, hole, domino, nor full sun. Suppose G contains an antihole of size at least 6. An antihole can only have two nonprobes, and deleting one of the nonprobes leaves at least the

complement of a P_5, a house, which is also a house in H. An antihole of size 5 is also a C_5.

Now we assume G contains a chordless cycle C with at least five vertices. From now on we can assume that $G = C$ and H is simply the embedding of C. If C has three or more consecutive probes, then there must be a hole in the embedding. Suppose then that the nonprobes of C are each followed by one or two probes before the next nonprobe. Say C contains the path $[a, b, c, d]$ with a and d nonprobes. Then a and d must be adjacent in H, or there will be a hole. Consider the graph H' obtained from H by deleting the four edges of $H[a, b, c, d]$. There must be a path from a to d in H'. If the distance from a to d in H' is 2, there is a house in H. If it is 3, there is a domino in H. If it is more than 3, there is a hole in H. Therefore, C consists of alternating probes and nonprobes.

Let C contain the path $G[a, b, c, d, e]$, where a, c, and e are nonprobes and suppose that the edges (a, c) and (c, e) are both lacking in H. If (a, e) is an edge in the embedding, we have a C_5. Otherwise let $[a = x_0, x_1, \ldots, x_i = e]$ be a chordless path in H which avoids b, c, and d. If x_1 is not adjacent to c in H, then the path $[c, b, a, x_1, x_2]$ is part of a chordless cycle in H of length at least 5. Therefore (c, x_1) is an edge. If (c, x_2) is an edge, then we have a house. If (c, x_2) is not an edge, but (c, x_3) is one (or if $i = 2$), then we have a domino. Otherwise the path $[c, x_1, x_2, x_3, x_4]$ is part of a hole (or $H[c, d, e, x_2, x_1]$ is a hole if $i = 3$ or $H[c, x_1, x_2, x_3, x_4]$ is a hole if c is adjacent to x_4). Therefore we know that G contains no P_5 with three nonprobes such that neither pair at distance 2 is adjacent in the embedding.

Assume then that C contains the path $G[a, b, c, d, e]$, and that (a, c) is an edge in the embedding, but (c, e) is not. If (a, e) is an edge we have a house. Otherwise consider a chordless path $[a = x_0, x_1, \ldots, x_i = e]$ in the embedding which avoids b, c, and d. If x_1 is not adjacent to c but x_2 is, we have a house with b as the roof. If neither x_1 nor x_2 is adjacent to c, then the path $[c, a, x_1, x_2, x_3]$ is part of a hole (or $H[a, c, d, e, x_1]$ is a hole if $i = 2$ or $H[c, a, x_1, x_2, x_3]$ is a hole if c is adjacent to x_3). Therefore x_1 is adjacent to c. Then we apply the same argument with a as the roof to get a house or hole unless x_2 is adjacent to c. Continuing inductively, we get that $x_i = e$ is adjacent to c, a contradiction. Thus, every pair of nonprobes at distance 2 in C is adjacent in H, and H must be a partial sun.

Since every trampoline contains a sun, H cannot be chordal, or there would be a full sun. If H were not even weakly chordal, then there would be a hole or a house. Therefore H contains at least a C_4. We choose a C_4 with three vertices “close" together. That is, we let a, b, c, and d be four nonprobes such that $H[a, b, c, d]$ is a C_4, and such that the unique path from a to c in $G - d$ contains b, and such that this path is the shortest possible for any choice

of a, b, c, and d such that $H[a,b,c,d]$ is a C_4. Clearly no two of a, b, c, and d are consecutive nonprobes in G (that is, at distance 2 in G); otherwise the intermediate probe forms a house with the C_4. Let u and w be the nonprobes immediately preceding and following b in G. Also relabel to vertices of the C_4 as v_1, v_2, v_3, and v_4, so that $b = v_1$ and the nonedges are (v_1, v_3) and (v_2, v_4).

First suppose that w is not adjacent to v_3. If w is adjacent to exactly one of v_2 and v_4, then $H[w, v_1, v_2, v_3, v_4]$ is a house. Suppose then that w is not adjacent to any of v_2, v_3, or v_4. Let $(w = w_1, w_2, \ldots, w_i)$ be the consecutive nonprobes in G after v_1 and suppose that $N(w_j) \cap \{v_2, v_3, v_4\} = \varnothing$ for $1 \leq j < i$ but $N(w_i) \cap \{v_2, v_3, v_4\} \neq \varnothing$. (Such a w_i must exist.) If $N(w_i)$ includes v_1 and at least one of v_2 or v_4, then we have a partial sun: the kernel consists of the vertices $(v_1, w_1, w_2, \ldots, w_i)$, and the independent set consists of the intermediate probes and v_2 or v_4. Then either the partial sun is chordal, and there is a full sun inside, or it is not chordal, and there is another C_4 induced by four vertices of the kernel. Since all four vertices of the new C_4 are on the path from b to c, they are too close together, contradicting the way we picked a, b, c, and d.

If $N(w_i) \cap \{v_1, v_2, v_3, v_4\}$ is $\{v_1, v_3\}$, $\{v_2, v_4\}$, or $\{v_2, v_3, v_4\}$, then we get another C_4 by replacing v_3 or v_4 by w_i, and v_1, w_1, and v_2 are closer than v_4, v_1, and v_2. In the other cases that $|N(w_i) \cap \{v_1, v_2, v_3, v_4\}| = 2$ we obtain a house. If w_i is adjacent to exactly one of v_2, v_3, or v_4, but not to v_1, we consider a chordless path from v_1 to w_i chosen from the vertices $(v_1, w_1, \ldots, w_i)$. Either we get a chordless cycle of length at least 5, or the path consists of v_1, w_i, and just one other vertex x, with w_i adjacent to v_2 or to v_4. In that case, $H[v_1, v_2, v_3, v_4, w_i, x]$ is a domino, a contradiction. We conclude that $w = w_1$ must be adjacent to both v_2 and v_4.

Again we let $(w_1, \ldots, w_i)$ be the nonprobes following v_1 in G, and suppose that w_i is adjacent to v_3 but that w_j is not adjacent to v_3 for $1 \leq j < i$. Such a w_i must exist, because if $H[w_{j-1}, v_2, v_3, v_4]$ is a C_4, and w_j is not adjacent to v_3, we just showed that w_j must be adjacent to v_2 and v_4, and $H[w_j, v_2, v_3, v_4]$ is also a C_4. If w_j happens to be the next nonprobe before $c \neq v_3$, then we get a house.

We also consider the nonprobes $(u = u_1, u_2, \ldots)$ on the other side of v_1. By the same argument, there must be some u_j which is adjacent to v_3, and the nonprobes $(u_1, u_2, \ldots, u_{j-1})$ are not adjacent to v_3. If u_j and w_i are not adjacent to each other, then we consider a chordless path from u_j to w_i on the vertices $(u_{j-1}, \ldots, u_1, v_1, w_1, \ldots, w_{i-1})$. With this path and v_3, either we have a hole, or we have a C_4 with three vertices in the interior of the original path from a to c, a contradiction. If u_j and w_i are adjacent, then the nonprobes $(u_j, \ldots, u_1, v_1, w_1, \ldots, w_i)$ form the kernel of a partial sun with the intermediate probes and v_3 as the independent set. Again we have a full

sun inside, or a C_4 contained within the original path from a to c. The proof is complete. □

Corollary 2.9. *Probe distance-hereditary graphs and probe interval graphs are weakly chordal.*

If $G = (\mathbb{P} + \mathbb{N}, E)$ is a probe perfect graph, then the complement of every odd cycle of length at least five contains exactly two nonprobes, and these are connected by an edge in any embedding of G. This observation, and the Probe Sandwich Theorem 2.3 mentioned above, led us to conjecture, somewhat boldly, the following:

Conjecture 2.10 (Partitioned Probe Perfect Graph Conjecture). There exists a polynomial time algorithm to test whether a partitioned graph $G = (\mathbb{P}+\mathbb{N}, E)$ is probe perfect.

Conjecture 2.11 (Probe Perfect Graph Conjecture). There exists a polynomial time algorithm to test whether a graph is probe perfect.

One of the merits of perfect graphs is that some of the 'basic' NP-complete problems such as CLIQUE, INDEPENDENT SET, CHROMATIC NUMBER, and CLIQUE COVER become solvable in polynomial time when restricted to this class [10]. For probe classes of perfect graphs, we have the following theorem:

Theorem 2.12. *Let $\mathcal{G}$ be any class of perfect graphs. Let $\mathcal{PG}$ be the class of partitioned probe graphs of $\mathcal{G}$. Then the* CLIQUE *problem can be solved in polynomial time for all graphs in $\mathcal{PG}$.*

Proof. Let $G = (\mathbb{P}+\mathbb{N}, E)$ be a partitioned graph in $\mathcal{PG}$. Recall that, if a graph is perfect the CLIQUE problem is tractable in polynomial-time via Lovász theta function; see, *e.g.*, pp. 325–356 in [10].
Observe that $\omega(G)$ can be computed as follows: For every vertex $x \in \mathbb{N}$, compute the maximum clique size of $G[N[x]]$, which is a perfect graph. Also compute the maximum clique size of $G[\mathbb{P}]$, which is likewise perfect. Then $\omega(G)$ will be the maximal value of these, and thus it can be computed in polynomial time. □

If $G = (\mathbb{P} + \mathbb{N}, E)$ is a partitioned probe perfect graph then its chromatic number is at most one more than $\omega(G[\mathbb{P}])$ since, after coloring the subgraph induced by the probes, at most one more color is needed to color the set of nonprobes. Thus for every partitioned probe perfect graph

$$\omega(G) \leq \theta(G) \leq \chi(G) \leq \omega(G[\mathbb{P}]) + 1 \leq \omega(G) + 1$$

The situation that hinders the settlement of the chromatic number occurs when $\theta(G) = \omega(G) = \omega(G[\mathbb{P}])$. In that case the chromatic number can be either $\omega(G)$ or $\omega(G) + 1$.

Conjecture 2.13. There exists a polynomial time algorithm to compute the chromatic number of a partitioned probe perfect graph.

Conjecture 2.14. There exists a polynomial time algorithm to compute the clique number of a probe perfect graph.

Conjecture 2.15. There exists a polynomial time algorithm to compute the independence number of a partitioned probe perfect graph.

Part II

Unpartitioned Classes

3

Self Complementary Classes

In this chapter we consider the recognition of probe graphs of some self-complementary classes. We show that there are polynomial-time recognition algorithms for (partitioned) probe cographs, P_4-reducible, P_4-sparse, and splitgraphs. The general strategy is to investigate the partitioned case first, and then to deal with the unpartitioned case by exhibiting a polynomial number of feasible partitions. We show that the probe graphs of all these classes are again perfect, thus by Theorem 2.6 on page 14 they are all classes of Meyniel graphs.

3.1 Probe cographs

A *cograph* is a graph without an induced P_4, *i.e.*, an induced path with 4 vertices [139]. Since the complement of P_4 is again P_4, it follows that cographs form a self-complementary class of graphs. By now there are many characterizations known and in the literature various characterizations of the class are used to define the class. They were discovered independently and given different names, for instance D^*-graphs, hereditary-Dacey graphs, 2-parity graphs, and complement-reducible graphs. For our purposes the following characterization will make the grade.

Theorem 3.1 ([50]). *Cographs can be characterized as follows:*

1. *A graph consisting of a single vertex is a cograph.*
2. *Let* G_1 *and* G_2 *be cographs. Then the* join *of* G_1 *and* G_2*, obtained by making every vertex of* G_1 *adjacent to every vertex of* G_2 *is again a cograph.*
3. *Let* G_1 *and* G_2 *be cographs. Then the (disjoint)* union *of* G_1 *and* G_2 *is again a cograph.*
4. *There are no other cographs.*

Notice that this decomposition recursively defines a *cotree* in which leaves correspond with the vertices of the graph and internal vertices are labeled as a *join* or a *union*. There is a wide variety of linear time cograph recognition algorithms. To mention just a few, see, *e.g.*, [27, 33, 52, 62, 99, 140].

To strike up an acquaintance with the class of probe cographs we mention some bagatelles.

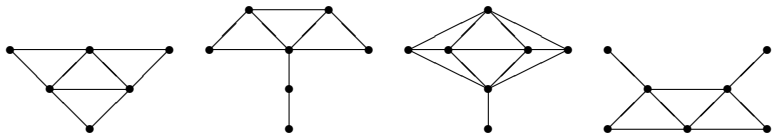


Fig. 3.1. The 3-sun, parapluie, parachute, and the co-rising sun.

Theorem 3.2. *Let* G *be a probe cograph. Then:*

a. *Every induced* $2K_2$[1] *in a probe cograph remains an induced* $2K_2$ *in every embedding.*
b. *If* G *contains an induced* P_5*, say* $P = [u, v, w, x, y]$*, then vertices* u*,* w*, and* y *must be nonprobes.*
c. G *has no induced* P_{k+1}*,* C_k *and* $\overline{C_k}$ *for any* $k \geq 5$*. Other forbidden induced subgraphs include the parachute, domino, co-rising sun, and the parapluie. We refer to Figures 3.1 and 8.1 on page 89 for depicting these graphs.*
d. *If* G *is a probe cograph then its* cliquewidth *is at most 4.*

Proof. (a) Notice that only one endpoint of each edge can be a non-probe. However, adding an edge between two endpoints of each edge induces a P_4.
(b) The only way to destroy all induced P_4s in P by joining nonprobes with edges is to add edges uw and wy. Therefore, u, w and y must be nonprobes.
(c) It immediately follows from (b) that a probe cograph cannot have an induced P_k and C_k for $k \geq 6$. A probe cograph cannot have an induced C_5 since such a cycle has at most two non-adjacent vertices and adding an edge between two such vertices cannot destroy all P_4's.
Since a maximal clique in a k-cycle ($k \geq 6$) has cardinality 2, the complement of such a cycle has independence number 2. Connecting such a pair of non-adjacent vertices cannot destroy all induced P_4's in the graph. Hence a probe cograph cannot have an induced $\overline{C_k}$ for $k \geq 5$.
Applying (a), it is easy to check that other forbidden induced subgraphs include the parachute, the domino, the co-rising sun, the parapluie, and plenty of others.

[1] That is, the disjoint union of two edges.

(d) Let $H = (V, E)$ be a cograph and let $(\mathbb{P}, \mathbb{N})$ be a partition of V such that G is obtained from H by making $\mathbb{N}$ an independent set. Recall that a graph is a cographs if and only if its cliquewidth is at most 2 (see, *e.g.*, [21, 90]). Consider a 2-expression for H. The two labels are used to construct H from two cographs H_1 and H_2 by taking the disjoint union or the complete join. We use two more labels, one for the non-probes in H_1 and one for the non-probes in H_2. □

Remark 3.3. Until now we have been unable to determine a complete list of forbidden induced subgraphs for the class of probe cographs.

Remark 3.4. Graphs of bounded cliquewidth allow polynomial time algorithms for many NP-complete problems [55]. A prerequisite, however, is that a cliquewidth expression of bounded width is available. Obtaining a cliquewidth expression for graphs with cliquewidth at most 4 is still an open problem.

Recall that a graph is *weakly chordal* if it has no induced C_k nor $\overline{C_k}$ for any $k \geq 5$ [105]. We have the following result.

Corollary 3.5. *A probe cograph is weakly chordal, hence perfect.*

Note that probe cographs form a *proper* subclass of the weakly chordal graphs. As an example may serve the rising sun, *i.e.*, the complement of the graph on the right in Figure 3.1.

The following observations will enable us to design a recognition algorithm.

Theorem 3.6. *Let G be a graph.*

1. *If G is disconnected then G is a probe cograph if and only if every component induces a probe cograph.*
2. *Assume $\overline{G}$ is disconnected and let $C_1, C_2, \ldots, C_k$ be the components of $\overline{G}$. Then G is a probe cograph if and only if all induced subgraphs $G[C_i]$, $i = 1, \ldots, k$ are cographs except possibly one which is a probe cograph.*
3. *Assume G and $\overline{G}$ be connected. If G is a probe cograph, then G^* is disconnected. If that is the case then, by Theorem 2.3 on page 13, G is a probe cograph if and only if every component of G^* induces a probe cograph in G.*

Proof. (1): Consider an embedding G' of G into a cograph. Let C be the vertex set of a component of G. Then $G'[C]$ is also a cograph since the class of cographs is hereditary. It follows that no edges need to be added between non-probes of different components of G.
(2): Suppose G is a probe cograph. Since the set of non-probes $\mathbb{N}$ is an independent set in G, it is a clique in $\overline{G}$. Thus at most one component of $\overline{G}$

contains non-probe vertices. Therefore all components except possibly one are necessarily cographs. If there is a non-cograph component then this must be a probe cograph.
Conversely, suppose $G[C_1]$ is a probe cograph and $G[C_i]$ is a cograph for $i \geq 2$. Consider an embedding of $G[C_1]$. Since G is the join of the subgraphs $G[C_i]$ for $i = 1, 2, \ldots, k$, we obtain an embedding of G into a cograph.
(3): Consider a cograph embedding H of G. Since G is connected, so is H. Since H is a cograph, $\overline{H}$ is disconnected, but $\overline{H}$ is a supergraph of G^*, hence also G^* is disconnected. □

Theorem 3.7. *There exists an $O(n^3)$ time algorithm to test whether a partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$ is a probe cograph.*

Proof. Let $G = (\mathbb{P}+\mathbb{N}, E)$ be a partitioned graph. We can use either linear time cograph recognition algorithm, *e.g.*, [52, 99] to test whether G is a cograph. If this is the case, we are done. Otherwise, assume G is disconnected and let $C_1, \ldots, C_k$ be the components of G, for some $k \geq 2$. By Theorem 3.6, G is a probe cograph if and only if each $G[C_i]$ is a probe cograph (with the induced partition into probes and nonprobes). We test recursively each $G[C_i]$ whether it is a probe cograph.
Assume $\overline{G}$ is disconnected with components $C_1, \ldots, C_k$. By Theorem 3.6, G is a probe cograph iff each $G[C_i]$ is a cograph except possibly one, which is a probe cograph and contains all nonprobes. Using an adjacency matrix for the graph G, we can find a representation for $\overline{G}$ and find the components of $\overline{G}$ in $O(n^2)$ time. There must be one C_i which contains all nonprobes. We test recursively whether $G[C_i]$ is a probe cograph. Other graphs $G[C_j]$ must be cographs, this can be tested in linear time.
Finally assume that G and $\overline{G}$ are connected. By Theorem 3.6, G^* is disconnected. Let $C_1, \ldots, C_k$ be the components of G^*. Then check whether each $G[C_i]$ is a probe cograph recursively with the induced partition.
We have described the recognition algorithm recursively. Its correctness follows from Theorem 3.6. The algorithm first checks whether the input graph is a cograph, this can be done in $O(n + m)$ time. If it is not, then it takes $O(n^2)$ time to compute the components $C_1, \ldots, C_k$ of G, $\overline{G}$, or G^*. By induction, each component C_i takes $O(|V(C_i)|^3)$ time. Notice that:

$$\sum_{i=1}^{k} c_i^3 \leq c_1^3 + (n - c_1)^3 \leq n^3 - 3nc_1(n - c_1) \leq n^3 - n^2$$

since $1 \leq c_1 \leq n - 1$, where we write $c_i = |V(C_i)|$. Since the overhead takes at most $O(n^2)$ time, this proves the theorem. □

Remark 3.8. The proof shows that each partitioned probe cograph G has a rooted decomposition tree where the leaves are the vertices of G and the

internal nodes are of *three* possible types: either a union node, when the subgraph induced by the leaves in the subtree is disconnected, or a join node when the complement of this subgraph is disconnected, or a "sandwich join" node when the subgraph and its complement are connected. This decomposition tree can be built in $O(n^3)$ time.

Remark 3.9. Alternatively, the general sandwich algorithm of [88] could be used. This algorithm runs in $O(n(n+m+|\mathbb{N}|^2))$ time.

The next theorem deals with the unpartitioned case and shows that there exists a polynomial number of feasible partitions.

Theorem 3.10. *Let* G *and* $\overline{G}$ *be connected and assume that* G *is not a cograph. Then* G *is a probe cograph if and only if there are two non-adjacent vertices* x *and* y *in* G *such that* G *is a probe cograph with probe set* $\mathbb{P} = N(x) + N(y)$ *and nonprobe set* $\mathbb{N} = V - \mathbb{P}$.

Proof. Assume G is a probe cograph with an embedding G'. We may assume that G' is the *join* of two graphs G'_1 and G'_2 since G' is obtained from the connected graph G by adding some edges.
Let $\mathbb{N}_i$ and $\mathbb{P}_i$ be the probes and nonprobes in G'_i, $i = 1, 2$. Since $\overline{G}$ is connected, we have that $\mathbb{N}_1 \neq \varnothing$ and $\mathbb{N}_2 \neq \varnothing$. Take any $x \in \mathbb{N}_1$ and $y \in \mathbb{N}_2$. □

Theorem 3.11. *The problem of recognizing probe cographs can be reduced to the problem of recognizing partitioned probe cographs in* $O(n+m)$ *time.*

Proof. If G is disconnected, then the problem reduces to the problem of testing whether each connected component of G induces a probe cographs in G. If $\overline{G}$ is disconnected, then all the nonprobes must lie in one component of $\overline{G}$. The problem reduces to the problem of testing whether each connected component of $\overline{G}$ is a cograph, except possibly one which is a probe cograph. Using the modular decomposition tree [146] of G, we can locate in linear time a set of modules which partition V, such that each module is connected and the complement is connected. The graph G is a probe cograph if and only if the graph $G[C]$ is a probe cograph for each such module C, with one additional restriction. For each module in the decomposition tree which induces a graph which is not coconnected, there can be only one coconnected component which is not a cograph. (This information can be read from the tree.)
In the following assume G and $\overline{G}$ are connected. First we run the cograph recognition algorithm of, *e.g.*, [52, 99] which tests if G is a cograph (we assumed it was not) and produces an induced P_4 in G if it is not. Let the induced P_4 found by the algorithm be $P = [a, b, c, d]$. We distinguish two possibilities.

Case 1. Assume $a, c \in \mathbb{N}$ and $b, d \in \mathbb{P}$. If $V(G) - (N(a) + N(c))$ is an independent set, then $N(a) + N(c)$ can be used as the set of probes. Otherwise consider an embedding H of G, which must be the join of two cographs H_1 and H_2. It is not hard to see that $\{a, b, c, d\} \subseteq V(H_1)$ or $\{a, b, c, d\} \subseteq V(H_2)$. Assume the former is the case. There must exist a nonprobe $\alpha \in V(H_2)$ since otherwise $\overline{G}$ would be disconnected. Then α is adjacent to b and to d and not adjacent to a nor c. Consider

$$\Omega = \{\alpha \mid [a, b, \alpha, d] \text{ is an induced } P_4 \text{ in } G\}$$

The vertices of Ω must all be nonprobes (or we have a P_4 we cannot destroy) and we have $\mathbb{P} = N(a) + N(\Omega)$. Note that this case includes the possibility that $\mathbb{P} = N(a) + N(c)$. Since Ω can be found in $O(n + m)$ time, so can $\mathbb{P}$. The feasible partition can be tested by an algorithm recognizing partitioned probe cographs.
The case where $a, c \in \mathbb{P}$ and $b, d \in \mathbb{N}$ is similar.

Case 2. Assume $a, d \in \mathbb{N}$ and $b, c \in \mathbb{P}$. Similarly to Case 1, if $N(a)+N(d)$ is not the complete set of probes, then the vertices $\{a, b, c, d\}$ cannot lie in both H_1 and H_2. Assume $\{a, b, c, d\} \subseteq V(H_1)$. There exists a nonprobe $\alpha \in V(H_2) \cap \mathbb{N}$. In G, α is adjacent to b and c and not adjacent to a and d. Now, it is easy to see that $\mathbb{P} = N(a) + N(\alpha)$. Find the set

$$\Omega = \{\alpha \mid b, c \in N(\alpha) \text{ and } a \notin N(\alpha) \text{ and } d \notin N(\alpha)\}$$

All vertices of Ω must be nonprobes, otherwise there exists a house in any embedding. Thus in this case $\mathbb{P} = N(a) + N(d) + N(\Omega)$. □

Corollary 3.12. *There exists an $O(n^3)$ algorithm for the recognition of probe cographs.*

An improvement of this algorithm will be described in Chapter 8.

3.2 Probe P_4-reducible graphs

P_4-reducible graphs were defined in [125].

Definition 3.13. *A graph G is* P_4-reducible *if every vertex belongs to at most one induced P_4 of G.*

Definition 3.14. *An* ornament *in a graph G is an induced P_4, $P = [a, b, c, d]$ such that every vertex of $G - V(P)$[2] is adjacent to b and to c and non-adjacent to a and to d.*

[2] Abusing notation we also use $G - P$ for the graph $G - V(P)$.

In [125] the following characterization of P_4-reducible graphs was given. It led to a linear time recognition algorithm for P_4-reducible graphs which appeared in [128].

Theorem 3.15 ([125]). *A graph G is P_4-reducible if and only if for every induced subgraph H of G, exactly one of the following conditions is satisfied:*

1. *either H or $\overline{H}$ is disconnected, or*
2. *there is a unique ornament.*

Remark 3.16. The class of P_4-reducible graphs is self-complementary.

Theorem 3.17. *Let G be a probe P_4-reducible graph.*

(1) *G is P_k-free for $k \geq 6$.*
(2) *G is C_k and $\overline{C_k}$-free for $k \geq 5$.*
(3) *G is weakly chordal hence perfect.*

Proof. (1): It's easy to verify that P_6 can not be embedded as a P_4-reducible graph. Hence a probe P_4-reducible graph has no induced P_k for $k \geq 6$.
(2): From (1) we see that G is C_k-free for $k \geq 7$. It can be verified that G is C_5- and C_6-free. $\overline{C_k}$ has independent number two. The resulting graph by connecting an edge of two independent vertices of $\overline{C_k}$ is still not P_4-reducible since any vertex incident to the two independent vertices is in at least two induced P_4s. Thus, G is $\overline{C_k}$-free for $k \geq 5$.
(3): Follows from (2) directly. □

Consider the recognition problem for probe P_4-reducible graphs. We first consider the partitioned case.

Lemma 3.18. *Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned probe graph. Assume P is a set of 4 vertices in G that can be made into an ornament of G by adding some edges between vertices of $\mathbb{N}$. Then G is probe P_4-reducible if and only if $G - P$ is probe P_4-reducible.*

Proof. Suppose that G is probe P_4-reducible. Since $G - P$ is an induced subgraph, $G - P$ is probe P_4-reducible.
If $G - P$ is probe P_4-reducible, then add edges incident with vertices of $P \cap \mathbb{N}$ and edges between vertices of $P \cap \mathbb{N}$ and $(G - P) \cap \mathbb{N}$ to make P an ornament. Also add edges between vertices of $\mathbb{N} - P$ to make $G - P$ P_4-reducible. It is easy to check that every vertex is then in at most one P_4 since each P_4 different from P is in $G - P$. □

The following lemma deals with the case where G or $\overline{G}$ is disconnected. The proof is identical to the proof of Theorem 3.6.

Lemma 3.19. *Let* G *be a graph.*

1. *If* G *is disconnected then* G *is probe* P_4*-reducible if and only if each component of* G *is probe* P_4*-reducible.*
2. *Let* $\overline{G}$ *be disconnected. Then* G *is probe* P_4*-reducible if and only if except one component which is probe* P_4*-reducible, all other components of* $\overline{G}$ *are* P_4*-reducible.*

The next lemma deals with the case that G is connected but the sandwich conjugate G^* is disconnected.

Lemma 3.20. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a connected partitioned graph. Let* C_i*,* $i = 1, \ldots, k$ *be the components of the sandwich conjugate* G^**, and let*

$$\mathbb{N}_i = C_i \cap \mathbb{N} \text{ and } \mathbb{P}_i = C_i \cap \mathbb{P}$$

Then G *is probe* P_4*-reducible if and only if each* $G[\mathbb{P}_i + \mathbb{N}_i]$ *is probe* P_4*-reducible with a vertex partition into probes* $\mathbb{P}_i$ *and nonprobes* $\mathbb{N}_i$*.*

Proof. If G is partitioned probe P_4-reducible, then each $G[\mathbb{P}_i + \mathbb{N}_i]$ is partitioned probe P_4-reducible since this is an induced subgraph.
Suppose that each $G_i = G[\mathbb{P}_i + \mathbb{N}_i]$ is probe P_4-reducible with vertex partition $(\mathbb{N}_i, \mathbb{P}_i)$. We can make an embedding of G by taking the join of the embeddings of the graphs G_i. □

Theorem 3.21. *There exists a polynomial time algorithm to test if a graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *is partitioned probe* P_4*-reducible.*

Proof. If G is disconnected, then G is probe P_4-reducible if and only if every component is probe P_4-reducible. If G is connected but G^* is not, then Lemma 3.20 applies.
Let $G = (\mathbb{P} + \mathbb{N}, E)$ and G^* be connected. If G is a partitioned probe P_4-reducible graph, let H be an embedding of G. Then H and $\overline{H}$ are connected. By Theorem 3.15, H has a unique ornament and Lemma 3.18 applies. This can be tested in polynomial time as there are at most $O(n^4)$ subsets P of 4 vertices. If neither lemma applies then G is not a partitioned probe P_4-reducible graph. □

Consider the recognition problem for unpartitioned probe P_4-reducible graphs. The proof of the following lemma can be copied from the proof of Theorem 3.10 so we omit it.

Lemma 3.22. *Let* G *and* $\overline{G}$ *be connected probe* P_4*-reducible graph with an embedding* G' *such that* $\overline{G'}$ *is disconnected. Then there are non-adjacent vertices* x *and* y *in* G *such that choosing* $\mathbb{P} = N(x) + N(y)$ *and* $\mathbb{N} = V - \mathbb{P}$ *provides a valid partition.*

We now describe the recognition algorithm for the unpartitioned case. Like in the algorithm for probe cographs, we try a polynomial number of feasible partitions of the vertex set into probes and nonprobes as input for the algorithm which checks the partitioned case.

i. Check the connectivity of G. If G is disconnected, and G is probe P_4-reducible if and only if every component is probe P_4-reducible. Henceforth assume that G is connected.
ii. Check the connectivity of $\overline{G}$. If $\overline{G}$ is disconnected with components $C_1, \ldots, C_k$, then G is probe P_4-reducible if and only if all induced subgraphs $G[C_1]$, $G[C_2], \ldots G[C_k]$ are P_4-reducible except one which is probe P_4-reducible. Henceforth assume that $\overline{G}$ is connected.
iii. Consider the case that there exists an embedding G' such that $\overline{G'}$ is disconnected. By Lemma 3.22 there must exist non-adjacent vertices x and y in G such that choosing $\mathbb{P} = N(x) + N(y)$ and $\mathbb{N} = V - \mathbb{P}$ provides a valid partition. In this step, try all such partitions until finding an embedding of G or go to next step.
iv. Consider the case that there is an embedding G' such that G' and $\overline{G'}$ are connected. Then by Theorem 3.15, G' has a unique ornament $P = [a, b, c, d]$. Since G is connected and a and d are pendant vertices, we may assume that $a, d \in \mathbb{P}$ in some valid partition if this exists. We consider three possibilities:
    a. There is a valid partition with $b \in \mathbb{P}$ and $c \in \mathbb{N}$. Hence b is adjacent in G to all vertices of $V - P$ and $\mathbb{N} = (V - P - N(c)) + c$.
    b. There is a valid partition with $b, c \in \mathbb{N}$. Hence b and c are not adjacent in G. Then $N(b) - P = N(c) - P = \mathbb{P} - \{a, d\}$.
    c. If $b, c \in \mathbb{P}$, then P is an ornament in G. In that case G is probe P_4-reducible if and only if $G - P$ is probe P_4-reducible.

By the discussion above we obtain:

Theorem 3.23. *It can be tested in polynomial time whether a graph G is probe P_4-reducible. If this is the case a valid embedding can be obtained in polynomial time.*

3.3 Probe P_4-sparse graphs

Hoàng introduced P_4-sparse graphs [111].

Definition 3.24. *A graph G is* P_4-sparse *if no set of 5 vertices induces more than one P_4.*

In [126] Jamison and Olariu characterized P_4-sparse graphs using spiders.

Definition 3.25. *A graph* G *is a* spider *if there is a partition of the vertices into three sets* S*,* K*, and* R*, satisfying:*

1. S *is an independent set,* K *is a clique, and* $|S| = |K| \geq 2$*,*
2. *every vertex of* R *is adjacent to every vertex of* K *and to no vertex of* S*,*
3. *there is a bijection* f *between* S *and* K *such that* either $\forall_{x \in S} N(x) = \{f(x)\}$*,* or $\forall_{x \in S} N(x) = K - f(x)$*. In the first case,* G *is called a* thin *spider and in the second case* G *is a* thick *spider.*

The set R *is called the* head *of the spider.*

Theorem 3.26 ([126]). *A graph is* P_4*-sparse if and only if for every induced subgraph* H *exactly one of the following conditions is satisfied:*

1. *either* H *or* $\overline{H}$ *is disconnected, or*
2. H *is isomorphic to a spider.*

Notice that the class of P_4-sparse graphs is self-complementary and properly contains the P_4-reducible graphs. Actually, a graph G is P_4-reducible if and only if G is P_4-sparse and $(S_3, \overline{S_3})$-free[126].[3] For a forbidden induced subgraph characterization of P_4-sparse graphs we refer to [126]. A decomposition tree for P_4-sparse graphs can be obtained in linear time [127].

Theorem 3.27. *Probe* P_4*-sparse graphs are perfect.*

Proof. Let G be a probe P_4-sparse graph. Note that C_5 contains more than one induced P_4. Any hole with more than 5 vertices has an induced P_5, hence it is not P_4-sparse. It follows that P_4-sparse graphs don't contain any hole. They also cannot contain a house (see Figure 8.1 on page 89), since a house has still two induced P_4s. Any cap, *i.e.*, a cycle with at least 5 vertices and exactly one chord between vertices at distance 2 in the cycle, contains a hole or is a house. Hence these are also not P_4-reducible. It follows that P_4-sparse graphs are Meyniel. By Theorem 2.6 on page 14 probe P_4-sparse graphs are perfect. □

Remark 3.28. Since C_6 is a probe P_4-sparse graph since it can be embedded as a 3-sun by making a triangle of an independent set with 3 vertices. Hence probe P_4-sparse graphs are not necessarily weakly chordal.

Notice that the statements and proofs of Lemmas 3.19, 3.20, and 3.22 can be carried over to similar statements and proofs for probe P_4-sparse graphs.

Lemma 3.29. *Let* G *be a graph.*

[3] The graph S_3 is the 3-sun, or Hajós graph. See Figure 3.1.

1. *If* G *is disconnected then* G *is probe* P_4*-sparse if and only if each component of* G *is probe* P_4*-sparse.*
2. *Assume that the complement* $\overline{G}$ *is disconnected. Let* $C_1, C_2, \ldots, C_k$ *be the vertex sets of the components of* $\overline{G}$*. Then* G *is probe* P_4*-sparse if and only if all induced subgraphs* $G[C_1], G[C_2], \ldots, G[C_k]$ *are* P_4*-sparse except possibly one which is probe* P_4*-sparse.*
3. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a connected, partitioned graph. Let* G^* *be the sandwich conjugate graph with connected components* $C_1, \ldots, C_k$*. Then* G *is probe* P_4*-sparse if and only if each* $G[C_i]$ *is probe* P_4*-sparse with the induced vertex partition into probes and nonprobes.*
4. *Assume* G *and* $\overline{G}$ *are connected probe* P_4*-sparse. Let* G' *be an embedding of* G *such that* $\overline{G'}$ *is disconnected. Then there exist non–adjacent vertices* x *and* y *in* G *such that* $\mathbb{P} = N(x) + N(y)$ *and* $\mathbb{N} = V - \mathbb{P}$ *is a valid partition.*

In the following lemma we show how to check whether a partitioned graph G can be embedded into a *thin spider* by adding some edges in $\mathbb{N}$.

Lemma 3.30 (Thin spider embedding). *Assume* $G = (\mathbb{P} + \mathbb{N}, E)$ *and* G^* *are connected.*

1. *Assume there exists a vertex* $x \in \mathbb{P}$ *and a* pendant *vertex* $y \in N(x)$ *such that* V *can be partitioned into a set* $S = y + (V - N[x])$ *of pendants,* $K = N(S)$*, and* $R = V - K - S$*, and assume that this partition can be completed into a thin spider by adding edges to* $\mathbb{N}$*. Then* G *is probe* P_4*-sparse if and only if* $G[R]$ *is probe* P_4*-sparse with the induced partition into probes and nonprobes.*
2. *Let* S *be the set of pendant vertices that are probes. Let* $K = N(S)$ *and* $R = V - K - S$*. Assume* G *can be embedded into a thin spider with this partition. Then* G *is probe* P_4*-sparse if and only if* $G[R]$ *is probe* P_4*-sparse with the induced partition.*
3. *Otherwise,* G *is not a probe thin spider.*

Proof. If G is a partitioned probe P_4-sparse, then $G[R]$ is probe P_4-sparse with the induced partition into probes and nonprobes since $G[R]$ is an induced subgraph of G.
Assume G can be completed into a thin spider with partition into sets S, K, and R. If there exists a vertex $x \in K \cap \mathbb{P}$, then let y be a pendant neighbor of x in S. The partition described in the first case occurs.
Assume that G can be embedded in a thin spider with $K \subseteq \mathbb{N}$. Since G is connected and $|K| \geq 2$, we have:

a. all pendant vertices in S are in $\mathbb{P}$,
b. $R \neq \varnothing$,
c. K has no pendant vertices, and
d. all pendant vertices in R are in $\mathbb{N}$.

Thus, in this case, S is exactly the set of pendant vertices in $\mathbb{P}$, $K = N(S)$, and $R = V - K - S$.
In both cases, if $G[R]$ is probe P_4-sparse, then add all edges to obtain an embedding of $G[R]$, and all edges to make G a thin spider. It is easy to see that any 5 vertices that induce a P_5 must be in R. Therefore, the embedding of G is P_4-sparse. □

Lemma 3.31 (Thick spider embedding). *Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned probe graph. Let G^* be the sandwich conjugate of G. Then G is a probe thick spider if and only if G^* is a probe thin spider.*

Proof. The class of P_4-sparse graphs is self-complementary. The complement of a thin spider is a thick spider. The claim now follows immediately from the Probe Sandwich Theorem 2.3. □

Theorem 3.32. *There exists a polynomial time algorithm that checks if a partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$ is probe P_4-sparse.*

Proof. If G is disconnected with components $C_1, \ldots, C_k$, then G is probe P_4-sparse if and only if every component $G[C_i]$ is probe P_4-sparse (with the induced partition). Likewise, by Lemma 3.29, item (3), we may as well assume that also G^* is connected. Thus we may assume that there is an embedding G' of G such that G' and $\overline{G'}$ are connected. By Theorem 3.26, G' is a spider. Now Lemmas 3.30 and 3.31 lead to a polynomial time algorithm to check whether G can be embedded into a thin or thick spider. □

Theorem 3.33. *There exists a polynomial time algorithm to test if a graph G is probe P_4-sparse and to produce a valid embedding if this is the case.*

Proof. The first two items of Lemma 3.29 deal with the case where G or $\overline{G}$ is disconnected. The seeking after an embedding G' such that $\overline{G'}$ is disconnected is easy by Lemma 3.29 item (4). This reduces the problem to the case where G can be embedded as a spider. By Lemma 3.31, we may further assume that G can be embedded as a thin spider. To do this, we let I be the set of pendant vertices. For each $y \in I$, there is a unique vertex x such that xy is an edge. Apply Lemma 3.30 to obtain a partition of V into sets S, K and R. If none of these partitions work, then all vertices in I must be probes. Let the set S in Lemma 3.30 be I. This will either reduce the graph G to the smaller subgraph R or allow the conclusion that G is not a probe P_4-sparse graph. Clearly only polynomial number of partitions need to be dealt with. □

3.4 Probe splitgraphs

Splitgraphs were introduced by Földes and Hammer in [73]. They are exactly the graphs G for which G and its complement are *chordal*.

Definition 3.34. *A graph is* chordal *if it has no induced cycle of length more than 3.*

Definition 3.35. *A graph* $G = (V, E)$ *is a* splitgraph *if its vertices can be partitioned into a clique* C *and an independent set* S*.*

We use $G = (C, S, E)$ to denote a splitgraph with clique C and independent set S. The following characterization of splitgraphs is on tap:

Theorem 3.36 ([73, 188]). *Let* G *be a graph. The following conditions are equivalent:*

(i) G *is a splitgraph.*
(ii) G *and* $\overline{G}$ *are chordal.*
(iii) G *has no induced* $2K_2$*,* C_4*, nor* C_5*.*

The fact that probe chordal graphs are perfect was observed in [89].

Theorem 3.37 ([89]). *Probe chordal graphs are perfect.*

Proof. An odd cycle C_{2k+1} has maximal independent sets of cardinality k. Choosing any independent set leaves at least one edge untouched which implies two consecutive probes. The two consecutive probes do not have a common neighbor, hence they must be in a chordless cycle of length at least 4 in any embedding. That is, the resulting graph is not chordal.
Removing an edge from C_k, $k \geq 6$ leaves at least two disjoint edges, *i.e.*, a C_4 in the complement. □

Corollary 3.38. *Probe splitgraphs are perfect.*

Remark 3.39. Notice that all bipartite graphs are probe splitgraphs. Besides the odd holes and odd antiholes, also the union of an edge and a K_3 is forbidden and there are plenty of others.

Assume that $G' = (C, S, E')$ is an embedding of a probe splitgraph $G = (V, E)$. Let $\mathbb{N}$ be the set of nonprobes, let $\mathbb{N}_C = \mathbb{N} \cap C$, and let $\mathbb{N}_S = \mathbb{N} \cap S$. Let $\mathbb{P} = V - \mathbb{N}$ and $\mathbb{P}_C = C - \mathbb{N}_C$. Notice that there must exist an embedding of G such that $\mathbb{N}_S = \varnothing$, for example an embedding with a minimal number of nonprobes, otherwise there won't be any embedding. Thus, to test whether an unpartitioned graph is a probe splitgraph, it is sufficient to hit upon $\mathbb{N}_C$ and add the necessary edges to turn it into a clique. Following our *modus operandi* we first consider the recognition problem for partitioned probe splitgraphs.

Theorem 3.40. *There is a* $O(n + m)$ *time algorithm to recognize partitioned probe splitgraphs.*

Proof. To boot the trivial case we test whether G is a splitgraph using some linear time algorithm, *e.g.*, the algorithm described in [103]. If G is a split graph we are done. Assume we are not that lucky.
Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned probe splitgraph and let $G' = (C, S, E')$ be an embedding of G. As affirmed above it suffices to find $\mathbb{N}_C$, add the necessary edges to turn it into a clique, and check whether the resulting graph is a split graph. We assume that G' is an embedding such that $|C|$ is maximal. It follows that a vertex of $\mathbb{N}$ belongs to S if and only if its degree in G is less than $|\mathbb{P}_C|$.
Use the algorithm of [103] to test whether $G[\mathbb{P}]$ is a split graph. If this is not the case then G is not a probe splitgraph. Otherwise, obtain a partition into clique C_P and independent set S_P of $G[\mathbb{P}]$, such that C_P is a maximal clique of $G[\mathbb{P}]$. Then $|\mathbb{P}_C|$ is either $|C_P|$ or $|C_P| - 1$.
Our algorithm computes, for the two possible values of $|\mathbb{P}_C|$, the sets $\mathbb{N}_S$ and $\mathbb{N}_C$ using the above degree condition. Then it makes $\mathbb{N}_C$ into a clique by adding edges and finally checks whether the resulting graph is a split graph. □

The unpartitioned case awaits.

Theorem 3.41. *A graph* G *is a probe splitgraph if and only if* G *is a splitgraph or if there exists a clique* K *and one vertex* $\kappa \in K$ *such that* $G - K$ *is bipartite and there is a bipartition of* $G - K$ *such that every vertex of one color class is adjacent to every vertex of* $K - \kappa$ *and not adjacent to* κ*.*

Proof. Assume G is not a splitgraph.
Let $G' = (C, S, E')$ be an embedding of G with a minimal number of nonprobes. Let $\mathbb{N}_C$ and $\mathbb{N}_S$ be the sets of nonprobes contained in C and S, respectively. Then $\mathbb{N}_S = \varnothing$. Hence $\mathbb{N}_C \neq \varnothing$. Picture a partition into C and S such that $|C|$ is maximal. Then every vertex of S has at most $|C| - 1$ neighbors in C. Consider the graph $G[\mathbb{N}_C + S]$. As clear as day, this subgraph is bipartite with color classes $\mathbb{N}_C$ and S. Let κ be any vertex in $\mathbb{N}_C$ and $K = (C - \mathbb{N}_C) + \kappa$. Then K is a clique in $N[\kappa]$, $G - K$ is bipartite with color classes $\mathbb{N}_C - \kappa$ and S, and every vertex of $\mathbb{N}_C - \kappa$ is adjacent to every vertex of $C - \mathbb{N}_C = K - \kappa$.
Now assume that the graph has a clique K with a specified vertex $\kappa \in K$ such that $G - K$ is bipartite with a bipartition such that every vertex of at least one of the 2 color classes is adjacent to every vertex of $K - \kappa$ and no vertex of this color class is adjacent to κ. We can make a clique of K together with this color class to complete the graph into a splitgraph. □

Notice that a vertex κ satisfying the conditions can be assumed to be a nonprobe in $\mathbb{N}_C$.

This characterization pimps the following algorithm:

Theorem 3.42. *There is a polynomial time algorithm which recognizes unpartitioned probe splitgraphs.*

Proof. Assume that G is not a splitgraph. Find one of the forbidden induced subgraphs; $2K_2$, C_4, or C_5. If the output is a C_5 then G is not a probe split graph by Lemma 3.37. Assume this is not the case and let H be the forbidden induced subgraph, *i.e.*, H is either $2K_2$ or C_4. Two vertices of H will be probes and the other two will be nonprobes and we may assume that these two nonprobes belong to $\mathbb{N}_C$.
For each of the four vertices of H the algorithm checks whether it is a vertex of $\mathbb{N}_C$ and can thus be chosen as the vertex κ in Theorem 3.41. This is done as follows for a fixed choice of vertex κ: Test whether $G[N[\kappa]]$ is a splitgraph in $O(n+m)$ time. If G is a probe splitgraph with $\kappa \in \mathbb{N}$ then $G[N[\kappa]]$ is a splitgraph. If $G[N[\kappa]]$ is not a splitgraph then reject the choice of κ. Otherwise find a partition (K_1, S_1) of $N[\kappa]$ such that K_1 is a clique and S_1 is an independent set in $G[N[\kappa]]$. Two cases pop up:

Case 1: $G[N[\kappa]]$ has a unique maximal clique. Then we may assume that K_1 is the unique maximal clique. Let $z \in K_1$ such that $|N(z)|$ is minimal. Then try $K = K_1$ or $K = K_1 - z$ as input for Theorem 3.41.

Case 2: $G[N[\kappa]]$ has at least two maximal cliques. In this case, we assume that K_1 is a minimal clique so that (K_1, S_1) is the split representation of $G[N[\kappa]]$. Let $z \in S_1$ with maximal degree. Then try $K = K_1$ or $K = K_1 + z$ as input for Theorem 3.41.

Consequently, for each fixed κ it takes $O(n + m)$ to obtain either one or two possible candidates for K and no other need to be verified. For each such K, it takes $O(n + m)$ time to check if K and κ meet the requirements of Theorem 3.41, which must be the case for some κ and some K if G is a probe splitgraph, and otherwise no such pair κ and K exists. The algorithm checks at most 3 vertices as candidates for κ, therefore, after the forbidden induced subgraph is found, the time complexity of the algorithm is linear. □

Remark 3.43. Recently Dieter Kratsch informed that he found a linear time algorithm that produces a $2K_2$, a C_4, or a C_5 if G is not a splitgraph [135]. If correct, the timebound in Theorem 3.42 reduces to linear time.

4

Probe Chordal Graphs

In this chapter we present algorithms to recognize partitioned and unpartitioned probe chordal graphs. These algorithms are based on Golumbic and Lipshteyn's original work [89] and they are described in Berry, Golumbic, and Lipsteyn's paper [14]. For the partitioned case they obtain an $O(nm)$ algorithm while for the unpartitioned case they obtain an $O(n^2m)$ algorithm. For a detailed time analysis we refer to [12].

4.1 Preliminaries

Recall the definition of a chordal graph:

Definition 4.1 ([100]). *A graph is* chordal *if it has no induced chordless cycle of length more than 3.*

Chordal graphs are amongst the most researched graph classes. Walter, in 1978, established the following characterization of chordal graphs which clarifies immediately the relation with trees. Given a family $\mathcal{S}$ of subtrees of a tree, a graph G is constructed in the following way. The vertices of G are the subtrees of $\mathcal{S}$, and two vertices are adjacent if the corresponding subtrees have at least one node of the underlying tree in common. For the following characterization see also [79, 29].

Theorem 4.2 ([190]). *A graph* G *is chordal if and only if* G *is the intersection graph of a family of subtrees of a tree.*

Notice that subtrees of a tree, seen as sets of tree nodes, fulfill the Helly property:

Definition 4.3. *Let* $\mathcal{E}$ *be a family of sets. Then* $\mathcal{E}$ *fulfills the* Helly property *if the following condition holds: If for any subfamily* $\mathcal{E}'$ *of* $\mathcal{E}$ *the elements of* $\mathcal{E}'$ *pairwise intersect then also* $\cap_{e\in\mathcal{E}'} e \neq \emptyset$.

Consider an intersection model for a chordal graph $G = (V, E)$ as a family of subtrees of a tree T. Notice that the smallest subtree containing a given leaf of T does not need to occupy more than exactly this leaf. The neighborhood of this vertex x is a clique in G. Using the Helly property, in a tree model T' for $G - x$, there is a node p in T' such that all vertices of $N(x)$ occupy this node. By extending the tree T' with one node adjacent to p, a tree model for G can be obtained. This proves by induction that T need not to be larger than the number of vertices of the graph.

For each node i in the tree, let S_i be the set of vertices of G of which the corresponding subtrees contain the node i. The sets S_i are usually called *bags*. The following properties 1-3 are obviously satisfied. The fourth follows from the Helly property and the fifth is a simple observation.

1. Every vertex $x \in V$ appears in at least one bag.
2. For every edge (x, y) in G, there exists a bag S_i such that $x, y \in S_i$.
3. For every vertex $x \in V$, the bags that contain x form a subtree of T, namely the subtree corresponding with x.
4. The *clique number* of G is the maximal size of a bag.
5. If G is a chordal graph then, by suitable contraction of edges in T, the nodes of T are in 1-1 correspondence with the maximal cliques of G. Hence the number of maximal cliques in a chordal graph is at most n.

As it turns out, the first three properties form exactly the definition of a tree decomposition of a graph, as defined by Robertson and Seymour. A *tree decomposition* for a graph G is a pair $(T, \mathcal{S})$ where T is a tree and $\mathcal{S}$ a set of subsets of V, which are in 1-1 correspondence with the nodes of T, such that the requirements 1-3 stated above are satisfied. They define the *width* of a tree decomposition as the maximal size of a bag minus one, and the *treewidth* of a graph as the minimum width over all possible tree decompositions of G. In this definition they subtract one in order to make a tree have treewidth 1. Notice the following:

> If G' is a subgraph of G, then the treewidth of G' is at most equal to the treewidth of G because any tree decomposition for G induces a tree decomposition for G'.

It follows that G has treewidth $\leq k$ if and only if G is a subgraph of a chordal graph with clique number at most $k+1$. The linear time recognition algorithm found by Bodlaender [19] for graphs of bounded treewidth, finds a tree decomposition, *i.e.*, an embedding of the graph in a chordal graph with bounded clique number. Since so many problems are well-known to be easily solvable for chordal graphs, especially when these have bounded clique number, it becomes clear that the same holds true for graphs of bounded treewidth. A

common approach is to perform dynamic programming on the subtrees and bags of the tree decomposition. Of course, exceptions to this rule are problems that are already NP-complete for trees, such as the bandwidth problem.

If G is a chordal graph and the tree nodes of the tree decompososition correspond uniquely with the maximal cliques in G, then this tree decomposition is also called a *clique tree* of G. Notice that the number of maximal cliques in G is at most n. It is also easy to see that the minimal separators in G are exactly the intersections of the bags that correspond with endvertices of edges in the clique tree of G [35, 77, 150], that is, every minimal separator is the intersection of two maximal cliques in G. The converse is another classic:

Theorem 4.4 ([66]). *A graph is chordal if and only if every minimal separator is a clique.*

Linear time algorithms for construction a clique tree can be found in [18, 183]. Other classical recognition algorithms for chordal graphs make use of perfect elimination orderings. The concept of a simplicial vertex was introduced by Lekkerkerker and Boland in [138].

Definition 4.5 ([138]). *A vertex* x *in a graph* G *is* simplicial *if* $N(x)$ *induces a clique in* G. *An ordering* $[x_1, \ldots, x_n]$ *of the vertices of* G *is a* perfect elimination ordering *if* x_i *is simplicial in* $G[\{x_i, \ldots, x_n\}]$ *for all* $i = 1, \ldots, n$.

The property of having a perfect elimination ordering characterizes chordal graphs:

Theorem 4.6 (Dirac's Theorem [66, 74, 174]). *A graph is chordal if and only if every induced subgraph has a simplicial vertex. Equivalently: a graph is chordal if and only if it has a perfect elimination ordering.*

If a chordal graph is not a clique, then it has at least two simplicial vertices which are not adjacent. This property allows one to find perfect elimination orderings of chordal graphs in linear time, for example using maximum cardinality search or lexicographic breadth first search [174, 186].

Paying their P's and Q's to Lekkerkerker and Boland, Berry *et al.* introduced the concept of an LB-simplicial vertex.

Definition 4.7. *A vertex* x *is* LB-simplicial *if for every component* C *of* $G - N[x]$, $N(C)$ *is a clique in* $G[N(x)]$.

Theorem 4.8 ([138]). *A graph is chordal if and only if every vertex is* LB*-simplicial.*

4.2 Partitioned probe chordal graphs

Recall that Meyniel graphs are those graphs in which every odd cycle of length at least 5 has at least two chords. Obviously, chordal graphs are Meyniel since, by definition, they don't have any chordless cycle of length more than 3. Hence probe chordal graphs are probe Meyniel. By Theorem 2.6 on page 14 probe chordal graphs are perfect. For a direct proof see Theorem 3.37 on page 33. In fact [89] contains the basic observation used in the recognition algorithm for probe chordal graphs:

> If $G = (\mathbb{P} + \mathbb{N}, E)$ is a partitioned probe chordal graph then there is no chordless cycle in G of length at least 4 with two adjacent probes.

We prove the converse of this observation in Theorem 4.11. We start with an analogue of Dirac's theorem. Recall our definition of a partitioned graph:

Definition 4.9. *A graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *with a partition of the vertices into a set* $\mathbb{P}$ *of* probes *and a set* $\mathbb{N}$ *of* nonprobes *is a* partitioned graph *if the set of nonprobes is an independent set.*

Definition 4.10. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph. A vertex* x *is* quasi-LB-simplicial *if for every component* C *of* $G - N[x]$*,* $N(C)$ *induces a partitioned probe clique in* $G[N(x)]$.

Theorem 4.11 ([14]). *Let* B *be a partitioned bipartite graph. The following conditions are equivalent:*

(i) G *is partitioned probe chordal bipartite.*
(ii) *Every probe vertex is quasi-LB-simplicial.*
(iii) *No induced chordless cycle of length at least 4 has two adjacent probes.*

Proof. **(i)** $\Rightarrow$ **(iii)**: If there is a chordless cycle C with at least 4 vertices and with two adjacent probes, then in any embedding of C the two probes will be in a chordless cycle of length at least 4, since they don't have a common neighbor in C.
(iii) $\Rightarrow$ **(ii)**: Let $x \in \mathbb{P}$. Assume there is a component C of $G - N[x]$ such that $N(C)$ has two vertices y and z which are not adjacent and of which at least one, say y, is a probe. Then consider a chordless y, z-path with internal vertices in C. Together with x, the path makes a chordless cycle with two adjacent probes, x and y.
(ii) $\Rightarrow$ **(i)**: We prove a somewhat more general claim. We prove that the claim holds for graphs with a partial fill-in of edges between nonprobes. We prove this by induction on the number of probes. First assume that G has no probes. Then we can make a clique of the nonprobes and obtain a chordal graph.

Now let x be a probe. By assumption x is quasi-LB-simplicial. For each component C of $G - N[x]$ make a clique of $N(C)$. Notice that this does not create any chordless cycles of length at least 4 with adjacent probes. Let G' be this graph. By induction $G' - x$ can be completed into a chordal graph by adding edges between nonprobes. Obviously, the same can be accomplished without adding edges between different components of $G' - N[x]$. Let H be such an embedding with x added back in. Every vertex of H is LB-simplicial hence, by Theorem 4.8, H is chordal. □

This wraps up the recognition for the partitioned case. It is easy to see that checking if every probe vertex is quasi-LB-simplicial takes only $O(nm)$ time, see *e.g.*, [12]. This proves:

Theorem 4.12 ([14]). *There exists an* $O(nm)$ *algorithm that checks if a partitioned graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *is probe chordal.*

Remark 4.13. It was shown in [12] that filling in the edges to make every vertex LB-simplicial takes only $O(nm)$ time. Hence an actual embedding can be obtained within the same time bound.

4.3 Probe chordal graphs

In this section we describe the $O(n^2m)$ algorithm of [14] for the recognition of (unpartitioned) probe chordal graphs.

Definition 4.14. *Let* G *be a graph.*

1. *A* C-edge *is an edge that is contained in a chordless cycle of length at least 4.*
2. *Two vertices are* C-equivalent *if they are connected by a path consisting of* C*-edges.*

Obviously, 'being C-equivalent' is an equivalence relation. The equivalence class are called C*-components*.

Definition 4.15. *A graph is* bicolorable *if there is a coloring of the vertices with two colors such that in every chordless cycle of length at least 4, no two adjacent vertices have the same color.*

Lemma 4.16 ([12]). *Let* Q *be a* C*-component. If* $G[Q]$ *is bicolorable then it has exactly two opposite bicolorings.*

Proof. Consider a bicoloring. Any pair of vertices $x, y \in Q$ is connected by a path P of C-edges. The colors of the vertices along P must alternate. Hence the color of any vertex determines uniquely the color of all other vertices in Q. □

Remark 4.17. Notice that $\overline{P_6}$ is not probe chordal. It is bicolorable, however every bicoloring produces an edge between vertices of the same color.

Lemma 4.18 ([12]). *Let* Q *be a* C*-component. Then* $G[Q]$ *is probe chordal if and only of one of the two colors in the bicoloring of* Q *induces an independent set in* G.

Proof. If $G[Q]$ is probe chordal then one of the two colors must be the set of nonprobes in any embedding. Conversely, if one of the two colors induces an independent set, then by Theorem 4.11 the graph $G[Q]$ is probe chordal. □

We need one more lemma for our recognition algorithm.

Lemma 4.19 ([12]). *Let* Q *and* R *be two bipartite* C*-components. Assume* Q *has a vertex* x *which is adjacent to at least two vertices in* R*. Then in any embedding* x *is a probe.*

Proof. Assume x is adjacent to an edge (a, b) in R. Since R is bipartite, one of a and b is a nonprobe in any embedding. Hence x must be a probe. If $N(x) \cap R$ is an independent set, then consider a shortest path between two neighbors of x. Together with x this path induces a chordless cycle of length at least 4, which contradicts $x \notin R$. □

In the rest of this chapter we assume that G is connected. It is easy to see that the collection of C-components can be obtained in polynomial time. See for example [181] which can be used to identify the C-edges in $O(nm^2)$ time in a straightforward fashion. For the optimal time analysis, which leads to $O(n^2m)$, we refer to [12, 14]. This step is the bottleneck of the algorithm's complexity. Assume we have obtained this collection of C-components. Proceed as follows:

Step 1. Find a bicoloring of every C-component. Check if at least one color induces an independent set in G. If this is not the case then STOP; the graph is not probe chordal. If there is exactly one color class which is independent, then assign this as a set of nonprobes. The remaining unlabeled C-components are bipartite.

Step 2. Check if there is no edge with both endvertices assigned as a nonprobe. If there is one, then STOP; the graph is not probe chordal.

Step 3. If there is an edge (x, y) and y is labeled as a nonprobe, then label x as a probe, and bicolor the C-component of x accordingly. Repeat the previous and this step until no more vertices are getting a label.

Step 4. If there is a vertex x in a C-component Q adjacent to two vertices in another C-component R, then check if x is already labeled as a nonprobe. If so; STOP; the graph is not a probe chordal graph. Otherwise, label x as

a probe, and label all vertices of Q accordingly. Repeat Step 3 and 4 until no more vertices are labeled or until the conclusion is drawn that G is not probe chordal.

Step 5. At this point the graph is probe chordal. To complete the labeling, first remove all C-components which are already labeled. Construct the quotient graph on the remaining C-components, by making two of these adjacent whenever there is an edge between them. This quotient graph is chordal. Consider a simplicial vertex Q. In Q there is at most one vertex x adjacent to vertices in other C-components. Label x as a probe and label the other vertices of Q accordingly. Remove Q from the quotient graph. Now notice that there must exist some C-component Q with at most one vertex x adjacent to vertices in other C-components. Label this vertex as a probe. Label the other vertices of Q accordingly, and remove Q.

This algorithm runs in polynomial time. We refer to [14] and [12] for a detailed time analysis. Their final result is:

Theorem 4.20 ([12]). *There exists an* $O(n^2m)$ *algorithm to recognize probe chordal graphs and to find an embedding if this is the case.*

Part III

Partitioned Classes

5

Partitioned Probe Chordal Bipartite Graphs

A bipartite graph is chordal bipartite if it has no induced cycles of length more than 4. We show that the recognition problem of partitioned probe chordal bipartite graphs is in P. For the class of partitioned probe strongly chordal graphs we have not been able to settle the complexity issue of finding an embedding. We discuss some partial results in the final section.

5.1 Preliminaries

Recall that a graph is *bipartite* if its set of vertices can be partitioned into two *color classes*, say X and Y, such that there are no edges between vertices contained in the same color class. We represent a bipartite graph as $B = (X, Y, E)$. A bipartite graph $B = (X, Y, E)$ is *complete bipartite* if every vertex of X is adjacent to every vertex of Y. If B is complete bipartite we also write $B = (X, Y)$.

Definition 5.1. *Let* $B = (X, Y, E)$ *be a bipartite graph. For an edge* $e = (x, y)$ *let* $N(e) = N(x) + N(y) - \{x, y\}$ *be the* neighborhood *of* e*. We write* $N[e] = N(x) + N(y)$ *for the* closed neighborhood *of* e*.*

Definition 5.2. *We call two edges* $e = (x, y)$ *and* $f = (a, b)$ distinct *if* $\{a, b\} \cap \{x, y\} = \varnothing$*. They are* disjoint *if they are distinct and* $a, b \notin N(e)$*.*

Definition 5.3. *A bipartite graph* $B = (X, Y, E)$ *is* chordal bipartite *if it does not have an induced cycle of length more than 4.*

Theorem 5.4. *A bipartite graph is chordal bipartite if and only if for every edge* $e = (x, y)$

1. $N[e] = V$ *or*

2. *for every component* C *of* $G - N[e]$, $N(C)$ *is complete bipartite.*

Proof. Assume B is chordal bipartite. Let C be a component of $B - N[e]$. If $N(C)$ has two nonadjacent vertices in different color classes then, with e and the adjacencies in C we find a chordless cycle of length at least 6. Conversely, if Ω is a chordless cycle of length at least 6, then any edge e of Ω has $\Omega - N[e]$ in some component C of $B - N[e]$ and $N(C)$ is not complete bipartite since the two neighbors of e in Ω are not adjacent. □

Theorem 5.4 can also be derived from various results on weakly chordal graphs [11, 105, 109]. More general results for hole-free graphs and weakly chordal graphs follow from a theorem by Chvátal, Rusu, and Sritharan.

Definition 5.5 ([41]). *A chordless path* $[x_1, \ldots, x_k]$ *is* simplicial *if it does not extend into any chordless path* $[x_0, x_1, \ldots, x_k, x_{k+1}]$.

Theorem 5.6 ([41]). *For every positive integer* k*, a graph contains no chordless cycle of length at least* $k+3$ *if and only if each of its nonempty induced subgraphs contains a simplicial path with at most* k *vertices.*

The case $k = 1$ is Dirac's classical result. From this result the following theorem is derived:

Theorem 5.7 ([107]). *A graph has no holes if and only if every induced subgraph either*

i. *has a star cutset,*
ii. *has a double star cutset, or*
iii. *is a clique or* C_4 *or two nonadjacent vertices.*

Notice that if a graph G has a (double) star cutset then it also contains a *full* (double) star cutset or otherwise a *dominated vertex*. A star cutset S centered at x is *full* if $S = N[x]$. A vertex x *dominates* a vertex y if $N(y) \subseteq N[x]$. If there is a star cutset S centered at x and if $N[x]$ is not a full star cutset then there is a component of $G - S$ completely contained in $N(x)$. Then x dominates all vertices of this component [107]. For chordal bipartite graphs we obtain the following (older) particularization.

Definition 5.8. *Let* $B = (X, Y, E)$ *be a bipartite graph. An edge* e *is* bisimplicial *if* $N[e]$ *induces a complete bipartite graph.*

Definition 5.9. *Let* $B = (X, Y, E)$ *be a bipartite graph and* $[e_1, \ldots, e_m]$ *an ordering of its edges. Let* $B_i = (X, Y, E_i)$ *with* $E_0 = E$ *and* $E_i = E_{i-1} - e_i$*. The ordering* $[e_1, \ldots, e_m]$ *is a* perfect edge elimination ordering *if* e_i *is bisimplicial in* B_{i-1}.

Theorem 5.10 ([87]). *A bipartite graph is chordal bipartite if and only if it has a perfect edge elimination ordering. Furthermore,* any *bisimplicial edge can be taken to start the ordering.*

In case of a perfect edge elimination ordering, a bisimplical edge is removed, but *not* its endvertices. If also the endvertices are removed, then the converse of Theorem 5.10 does not hold. The *6-pan*, *i.e.*, a 6-cycle plus one pendant vertex, is clearly not chordal bipartite. It has a bisimplicial edge, namely the edge incident with the pendant vertex. If the bisimplical edge is removed *together* with its endvertices, we end up with a P_5, which is chordal bipartite.

The greedy algorithm which takes any bisimplicial edge to start with can be used to find a perfect edge elimination ordering. Efficient algorithms that find a perfect edge elimination ordering appeared for example in [81, 133]. In [81] it is shown that a bisimplicial edge in a chordal bipartite graph can be found in $O(n^2)$ time. Hence a perfect edge elimination ordering for a chordal bipartite graph can be obtained in $O(n^2m)$ time.

Since chordal bipartite graphs are bipartite weakly chordal graphs we shortly review some older noteworthy ideas that play a role for this class. Hayward [105] introduced the class of weakly chordal graphs as those graphs having no holes or antiholes. Obviously, these graphs are perfect. The up-to-date fastest recognition algorithms for weakly chordal graphs are [11, 108, 164]. All these algorithms take $O(n + m^2)$ time. The algorithm of [108] uses only linear space while that of [11] uses $O(n + m^2)$ space and that of [164] uses $O(nm)$ space. If the graph has no induced C_5 the space complexity of the last algorithm can be reduced to $O(n + m)$. Only the algorithm of [164] can be used to find resolutely holes or antiholes in graphs. The algorithm of [108] produces only one of the two, while the algorithm of [11] in some cases terminates producing neither a hole nor an antihole.

Definition 5.11 ([109]). *Two vertices* x *and* y *in a graph* G *form a* 2-pair *if each chordless path between them has exactly two edges. A* co-pair *is a 2-pair in* $\overline{G}$.

Observe that x and y form a 2-pair if and only if the common neighborhood of x and y is a separator of the graph. The fastest way to find a 2-pair, as far as we know, is via an $O(nm)$ time algorithm [3] or, alternatively, via fast matrix multiplication in $O(n^{2.376})$ time.[1]

Theorem 5.12 ([109]). *A graph* G *is weakly chordal if and only if every induced subgraph of* G *either is a clique or else has a 2-pair.*

[1] As far as we know Coppersmith and Winograd's matrix multiplication is the currently fastest [47]. The conjecture is that $\Theta(n^2)$ suffices.

Theorem 5.13 ([109, 184]). *Let G be a graph with a 2-pair x, y, and let G' be the graph obtained from G by either adding the edge (x, y) to G or identifying them to form a single vertex. Then G is weakly chordal if and only if G' is weakly chordal.*

A general idea for a recognition algorithm of weakly chordal graphs is to look for a 2-pair [3, 184]. Connect the two vertices that form a 2-pair and repeat the search for a 2-pair. The graph is weakly chordal if and only if this procedure ends with a clique. This characterization led to an $O(n^4)$ time recognition algorithm [184]. Furthermore, Theorem 5.13 makes it possible to think up efficient algorithms for NP-complete problems such as CLIQUE, INDEPENDENT SET, CHROMATIC NUMBER, and CLIQUE COVER when restricted to the class of weakly chordal graphs [108, 109]. As already mentioned above, the time complexity for the recognition problem was improved in [11, 108, 164, 179] to $O(n + m^2)$ time. Weakly chordal graphs have at most $n + m$ minimal separators and they can be listed in $O(n+m^2)$ time [11]. The procedure described above also generates all minimal separators of a weakly chordal graphs.

A characterization generalizing Theorem 5.4 for weakly chordal graphs is the following.

Definition 5.14. *An edge e is* LB*-simplicial if one of the following holds:*

1. $N[e] = V$ *or*
2. *for every component C of $G - N[e]$, and for every component Δ of $\overline{G}[N(C)]$ at least one endvertex of e is adjacent to all vertices of Δ.*

Theorem 5.15 ([11]). *A graph is weakly chordal if and only if every edge is* LB*-simplicial.*

To illustrate the relation between chordal bipartite graphs and strongly chordal graphs we add the following section on totally balanced matrices.

5.1.1 Totally balanced matrices

Recall that a graph is *strongly chordal* if it is chordal and every even cycle of length at 6 has an odd chord [71]. Chordal bipartite graphs are the bipartite analogue of the strongly chordal graphs. They were authoritatively introduced in [87]. The close connection is best explained via the concept of a totally balanced matrix. Let A be an $m \times n$ $0,1$-matrix, let $\mathbf{b}$ be a $0,1$-vector. Let $P(A, \mathbf{b})$ and $Q(A, \mathbf{b})$ denote the following two polyhedra.

$$P(A, \mathbf{b}) = \{\mathbf{x} \in \mathfrak{R}^n \mid A\mathbf{x} \leq \mathbf{b} \text{ and } \mathbf{x} \geq 0\}$$
$$Q(A, \mathbf{b}) = \{\mathbf{x} \in \mathfrak{R}^n \mid A\mathbf{x} \geq \mathbf{b} \text{ and } \mathbf{x} \geq 0\}$$

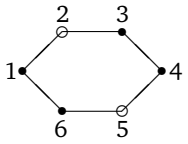
A $0,1$-matrix A is *perfect* if the polyhedron $P(A,\mathbf{b})$ has only integral extrema for every $0,1$-vector $\mathbf{b}$. A $0,1$-matrix is *balanced* if $Q(A,\mathbf{b})$ has only integral extrema for every $0,1$-vector $\mathbf{b}$. Balanced matrices are perfect [75]. When the matrix A of a linear programming problem is balanced then many optimization problems have a polynomial time solution thanks to the fact that the two polyhedra $P(A,\mathbf{b})$ and $Q(A,\mathbf{b})$ *both* have only integral extremal points [75] for every $0,1$-vector $\mathbf{b}$. Berge proved that a $0,1$-matrix is balanced precisely when it has no square submatrix of odd order with precisely two ones in each row and column [7, 80].[2] When viewed as the adjacency matrix of a bipartite graph where columns and rows correspond to the vertices of the two color classes, the balanced matrices correspond uniquely to the bipartite graphs with no holes of length $2 \mod 4$. These bipartite graphs are called *balanced*. Conforti, Cornuéjols, and Rao designed polynomial recognition algorithms for balanced matrices [46, 107]. They proved that every balanced bipartite graph is either *totally unimodular* or has a double star cutset. A $0,1$-matrix A is totally unimodular if the determinant of every square submatrix is 0, 1, or -1, that is, if $P(A,\mathbf{b})$ is integral for *all* integral $\mathbf{b}$ [117, 167]. (See also [65]). Seymour [185] proved a decomposition theorem (using 2-joins) for totally unimodular matrices which is the basis for a recognition algorithm. A bipartite graph is totally unimodular if the corresponding matrix, as defined above is totally unimodular.

It can be seen that a graph is perfect if and only if its *clique matrix* is perfect [40]. A *graph* is balanced if its clique matrix is balanced; so balanced graphs are perfect [8]. If a connected graph is balanced then the number of maximal cliques is at most the number of edges [22]. Given a graph, its maximal cliques can be listed with polynomial delay in $O(nmk)$ time, where k is the number of maximal cliques [187]; hence, there is a polynomial-time recognition algorithm for balanced graphs [63]. A characterization in terms of "extended odd suns" appeared in [22].

[2] Berge called a $0,1$-matrix *bicolorable* if the columns can be partitioned into red and blue such that every row with at least two 1s contains a 1 in a blue and a 1 in a red column. He proved that a $0,1$-matrix is balanced if and only if every submatrix is bicolorable. A bicoloring algorithm is given in [32]. This algorithm produces either a bicoloring or a chordless cycle of length $2 \mod 4$ in the bipartite graph. Notice that this algorithm does *not* provide a test for a matrix to be balanced. The bicoloring-concept is extended in [44] in order to characterize balanced $0,\pm1$-matrices. A similar characterization for totally unimodular $0,1$-matrices (and also for $0,\pm1$-matrices) was obtained by Ghouilla-Houri [80]. (See also [167].) A $0,1$-matrix is totally unimodular if and only if for every subset of the columns there exists a partition into red and blue columns such that for every row the number of red and blue ones differs at most one.

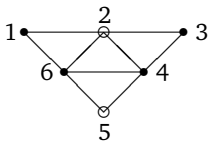
Lovász [142] defined a hypergraph to be *totally balanced* if every cycle of length greater that 2 has some edge incident with at least three vertices of the cycle. Based upon this definition we have:

Definition 5.16 ([1, 2]). *A* $(0,1)$*-matrix is* totally balanced *if it does not contain as a submatrix the vertex-edge incidence matrix of a cycle of length at least 3. Equivalently, a* $0,1$*-matrix is* totally balanced *if the bipartite graph obtained by letting rows and columns represent the vertices of the two color classes and by making two vertices adjacent if the matrix has a one in the corresponding row and column, is chordal bipartite.*



$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ \square & 1 & 1 \end{pmatrix}$$

Fig. 5.1. A 6-cycle. Vertices 2 and 5 are nonprobes. In order to make the bipartite graph (totally) balanced (*i.e.*, chordal bipartite), the edge between 2 and 5 has to be added. In the matrix, with rows labeled $[1,3,5]$ and columns labeled $[2,4,6]$, this entry has to be filled with a 1.



$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & \square & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & \square & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & \square & 1 & 1 \\ 0 & 0 & \square & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Fig. 5.2. A labeled 3-sun. Vertices 2 and 5 are nonprobes. The other vertices are probes. The first matrix is the closed neighborhood matrix with vertex ordering $[1,2,3,4,5,6]$ where 'feasible edges' between nonprobes are replaced by $\square$s. The second is a symmetric doubly lexical ordering (with ordering $0 < \square < 1$) of this matrix with vertex ordering $[3,1,2,5,4,6]$. The third matrix shows a Γ-free ordering $[3,1,5,4,6,2]$ with the $\square$s replaced with 1s. Hence this embedding is strongly chordal. Notice that when the $\square$s are replaced with 0s in the second matrix it is still doubly lexical ordered but it contains a Γ. Notice that the completion problem of a symmetric $0,1,\square$-matrix with 1s on the diagonal, into a symmetric totally balanced matrix is precisely the strongly chordal sandwich problem [88]. The complexity of this problem is open.

A graph is *totally balanced* if its clique matrix is totally balanced. The class of totally balanced graphs is exactly the class of the strongly chordal graphs:

a graph is strongly chordal if and only if its clique matrix is totally balanced [71, 143]. The *closed neighborhood matrix* is obtained from the ordinary adjacency matrix by putting 1's on the diagonal. For every graph, the clique matrix is totally balanced if and only if its closed neighborhood matrix is totally balanced [71]. Hence:

Theorem 5.17 ([71]). *A graph is strongly chordal if and only if its closed neighborhood matrix is totally balanced. A graph is chordal bipartite if and only if its adjacency matrix is totally balanced.*

The following concept was introduced for the recognition of strongly chordal graphs, independently in [116, 143]. A *lexical ordering* of vectors is the standard lexicographic ordering except that the vectors are read backwards, *i.e.*, from highest to lowest coordinate. A *doubly lexical ordering* of a real-valued matrix is an ordering of its rows and columns such that the rows and also the columns as vectors are lexically increasing. Notice that every real-valued matrix has a doubly lexical ordering [143]. If a symmetric matrix has a dominant diagonal,[3] such as the closed neighborhood matrix of a graph, then there also exists a symmetric doubly lexical ordering [143]. For a $k \times \ell$ matrix, a doubly lexical ordering can be found in $O(L \log(k+\ell) + k + \ell)$, where $L = k + \ell + f$ and f is the number of entries not equal to the smallest entry [168, 182].

Definition 5.18. *A* Γ *is a submatrix of the form* $\left(\begin{smallmatrix} 1 & 1 \\ 1 & 0 \end{smallmatrix}\right)$.

Theorem 5.19 ([143]). *A* $(0,1)$*-matrix has a* Γ*-free ordering if and only if it is totally balanced. A doubly lexical ordering of a totally balanced matrix is* Γ*-free.*

This characterization leads to an algorithm to recognize strongly chordal graphs and chordal bipartite graphs in $O(\min\{n^2, m \log n\})$ time [71, 143, 168, 182].

5.1.2 Biclique trees

We end these introductory sections with an analogue of clique trees for chordal graphs. A chordal graph is the intersection graph of a family of subtrees of a tree [79, 190]. Recall that a *clique tree* for a chordal graph G is a tree T together with a 1-1 mapping from the maximal cliques in G to the nodes in T such that the maximal cliques in G that contain any given vertex x are mapped to a subtree of T. For chordal bipartite graphs we obtain a similar

[3] The diagonal of a symmetric matrix A is *dominant* if $A(i,i) \geq A(i,j)$ for all indices i and j.

result. Recall that a chordal graph can have at most n maximal cliques. Similarly, using Theorem 5.10 it is easy to see that a chordal bipartite graph can have at most m maximal bicliques (see, e.g., Lemma 8.2.1 in [132]):

Definition 5.20. *A* biclique *in a bipartite graph* $B = (X, Y, E)$ *is a complete bipartite subgraph that contains at least one* C_4.

For a biclique C in a bipartite graph $B = (X, Y, E)$, with color classes $P \subseteq X$ and $Q \subseteq Y$ we write $C = (P, Q)$.

Definition 5.21. *Let* $B = (X, Y, E)$ *be chordal bipartite. A* biclique tree *is a tree* T *together with a 1-1 mapping from the maximal bicliques in* B *to the nodes in* T *such that for every edge, the maximal bicliques that contain that edge are mapped to a subtree of* T.[4]

For the following result, see also [28, 152, 154].

Theorem 5.22. *Every chordal bipartite graph* B *has a biclique tree* T. *If* B *is biconnected, every vertex is contained in bicliques that form a subtree.*

Proof. Assume B is a chordal bipartite graph. If B contains no C_4 we take an empty tree. If B is a biclique then we can take a tree T with one node and map B to this node.
Assume B is not a biclique. Consider a bisimplicial edge $e = (x, y)$. If $N(x) = y$ or $N(y) = x$, then e is not contained in any biclique. In that case a biclique tree for $B - e$ will serve as a biblique tree for B.
Assume $N(x) - y$ and $N(y) - x$ are not empty. Let $C = (N(x), N(y))$. Assume $C - \{x, y\}$ is contained in a biclique of $B - \{x, y\}$. Then consider $B - \{x, y\}$. This graph is chordal bipartite hence there is a biclique tree T'. Consider a maximal biclique C' that contains $C - \{x, y\}$. Assume C' is mapped to a node p in T'. Create a new node q and make this adjacent to p. Map C to q. Since all edges incident with x or y are in C only and all other edges of C are in C', the new tree T is a biclique tree for B.
Assume $C - \{x, y\}$ is not contained in a biclique of $B - \{x, y\}$. Then x and y each have only one neighbor y' and x' respectively and these are adjacent, i.e., $C = (\{x, x'\}, \{y, y'\})$. Construct a biclique tree T' for $B - \{x, y\}$. If (x', y') is not contained in any biclique of $B - \{x, y\}$, then make a new node q adjacent to any node in T' and map C to q. Assume (x', y') is contained in some maximal bicliques of $B - \{x, y\}$. Choose any of these and let it be mapped to a node p of T'. Make a new node q adjacent to p and map C to q.
Assume that B is biconnected. We claim that every *vertex* is in bicliques that form a subtree. Assume not. Assume x is in some node p and in q but not in

[4] Abusing notation somewhat we also call the tree T a biclique tree.

any node on the p, q-path in T. There are edges $e_p \in p$ and $e_q \in q$ incident with x, but with distinct other endvertices. Since B is biconnected, the other endvertices must be in some path that does nor contain x. If the edges e_p and e_q are in a C_4, then x must appear also in the nodes on the p, q-path. Otherwise we find a chordless cycle of length at least 6. □

5.2 Partitioned probe chordal bipartite graphs

Chordal bipartite graphs are exactly the bipartite weakly chordal graphs.[5]

In this section let $B = (X, Y, E)$ be a bipartite graph with a partition of the vertices into a set $\mathbb{P}$ of probes and an independent set $\mathbb{N}$ of nonprobes. For a subset Z of vertices we write $\mathbb{P}(Z) = \mathbb{P} \cap Z$ and $\mathbb{N}(Z) = \mathbb{N} \cap Z$.

Definition 5.23. *A* feasible edge *is either an edge of* B *or two nonprobes.*

Definition 5.24. *For a feasible edge* e *and a component* C *of* $B - N[e]$ *such that* $N(C) + e$ *is probe complete bipartite, the* block (e, C) *is the subgraph of* B *induced by* $e + C + N(C)$.

For a feasible edge e with vertices x and y we use the notation $e = (x, y)$ and call x and y the endvertices of e.

Lemma 5.25. *If* B *is chordal bipartite,* (e, C) *a block such that* $N(C)$ *has at least one vertex in each color class, then there exists an edge* e' *in* C *such that* $N(C) \subseteq N(e')$.

Proof. Two vertices in one color class having private neighbors in C imply a chordless cycle of length at least 6 (with x or y). Hence there exists vertices x' and y' in the two color classes such that $N(C) \subseteq N(x') + N(y')$. Consider a chordless x', y'-path in C. This shows that there must exist an edge. □

Lemma 5.26. *Let* (e, C) *be a block with vertex set* T. *There exists an embedding* B^* *for* $B[T]$ *such that* e *is an edge and* $N_{B^*}(C)$ *is complete bipartite if and only if either* $B(T)$ *is probe complete bipartite or there exists a feasible edge* $e' = (x', y')$ *in* C *such that:*

1. $N(C) \subseteq N(e')$ *and*
2. *for every component* C' *of* $B - N[e']$ *that does not contain* e *there exists an embedding of the block* (e', C') *such that* e' *is an edge.*

[5] It yields to argument that they should be called this way, since chordal bipartite graphs are *not* chordal. Other names, like *bichordal* have been suggested, *e.g.*, in [169]. To avoid further discussion we don't suggest *totally balanced graphs*.

Proof. Consider an embedding B^* of $B[T]$ with e an edge and $N_{B^*}(C)$ complete bipartite. By Theorem 5.4, in the embedding either $N_{B^*}[e] = T$ or for every component C' of $T^* - N_{B^*}[e]$, $N_{B^*}(C')$ is complete bipartite. In the first case x and y are made adjacent to all vertices of their opposite color classes in C. Hence C contains only nonprobes and since $\mathbb{N}$ is an independent set and $B[C]$ is connected, $|C| = 1$. Hence $B(T)$ is probe complete bipartite. Consider the second case. We claim that there exists an alternative embedding B' obtained from B^* by deleting all edges in B^* from e to C. Assume B' has a chordless cycle of length at least 6. Then either x or y, say x makes a chord with some vertex c in this cycle in B^*. Then the neighbors of x must have private neighbors in C, which is a contradiction.

□

5.3 Partitioned probe strongly chordal graphs

Strongly chordal graphs are those chordal graphs without an induced sun. They are strongly related to chordal bipartite graphs. So far we have not been able to device a polynomial-time algorithm for the recognition of partitioned probe strongly chordal graphs. In this section we mention some partial results and some open problems.

Farber [71] introduced strongly chordal graphs. His motivation was a polynomial time algorithm for the WEIGHTED DOMINATING SET problem on strongly chordal graphs, which is NP-complete for chordal graphs in general. Recall that a *chord* in a cycle is an edge connecting two vertices of the cycle which are not adjacent in the cycle.

Definition 5.27. *An* odd chord *in a cycle of even length at least 6 is a chord joining two vertices whose distance along the cycle is odd.*

Definition 5.28. *A graph is* strongly chordal *if and only if it is chordal and every even cycle of even length at least 6 has an odd chord.*

Actually, Farber [71] defined strongly chordal graphs as those graphs that have a *strong elimination ordering*. The above is an equivalent characterization [71, Theorem 6.1].

Recall that chordal graphs are characterized by the presence of a simplicial vertex in every induced subgraph [66]. For strongly chordal graphs there is a similar characterization.

Definition 5.29. *A* simple vertex *is a vertex x such that for every pair $y, z \in N(x)$ either $N[y] \subseteq N[z]$ or $N[z] \subseteq N[y]$.*

That is, the closed neighborhoods of the neighbors of a simple vertex form a chain under inclusion. Notice that simple vertices are simplicial. Farber proved the following analogue of Dirac's Theorem (4.6 on page 39):

Theorem 5.30 ([71]). *A graph is strongly chordal if and only if every induced subgraph has a simple vertex.*

Since the class of strongly chordal graphs is hereditary every strongly chordal graph has a *simple elimination ordering*, which is the analogue of a perfect elimination ordering for chordal graphs. That the appearance of a simple elimination ordering characterizes strongly graphs is true but not obvious. In his paper Farber proves that if every even cycle of length at least 6 has an odd chord then every induced subgraph has a simple vertex. In order to do this he makes use of a forbidden induced subgraph characterization.

Definition 5.31. *A* sun *is a graph obtained from an even cycle of length at least 6 in which edges are added to make a maximum independent set into a clique.*

If the cycle is of length $2k$, then the sun is called a k-*sun*. A 3-sun is illustrated in Figure 3.1 on page 22. If some edges of the clique are possibly missing, but the graph is still chordal, it is called a *trampoline*. We call the set of vertices of degree more than 2 the *kernel* of the trampoline. Farber and Chang [71, Lemma 4.5] observed that every trampoline has a sun as an induced subgraph. Notice that a strongly chordal graph does not have an induced sun, since the even cycle has no odd chord.

Theorem 5.32 (Farber's Theorem [34, 71]). *A graph is strongly chordal if and only if it is chordal and has no induced sun.*

Theorem 5.33 ([71]). *A graph is strongly chordal if and only if it has a simple elimination ordering.*

The following theorem follows also from Theorem 2.8 on page 14.

Theorem 5.34. *Probe strongly chordal graphs are weakly chordal.*

Proof. Since probe strongly chordal graphs are probe chordal, they are perfect (see Theorem 3.37 on page 33). Consider a hole of length $2k$, $k \geq 3$. In order to make this chordal one needs a set of nonprobes of size k. This can be made a sun by turning it into a clique. Since every trampoline contains a sun as an induced subgraph [34, 71, 123] the embedding will have a sun.
Any path of length at least 6 has two disjoint edges. Hence, deleting any edge from a hole of length at least 6 leaves a 4-cycle in the complement. □

Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned graph. Let $C = [a, b, c, d]$ be a chordless 4-cycle in G such that probes and nonprobes alternate in C. Any chordal embedding of G must have the edge filled in between the nonprobes in C.

Definition 5.35 ([89]). *The* enhanced graph G^* *is the graph obtained from a partitioned graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *by adding all edges between nonprobes in alternating 4-cycles of* G.

Golumbic and Lipshteyn obtained the following result for partitioned probe chordal graphs which are weakly chordal:

Theorem 5.36 ([89]). *Let* $G = (\mathbb{P}+\mathbb{N}, E)$ *be a partitioned probe chordal graph which is weakly chordal. Then the enhanced graph* G^* *is chordal.*

Unfortunately, this does not solve the recognition problem for partitioned probe strongly chordal graphs. As an example take the 3-sun (see Figure 3.1 on page 22) in which one simplicial vertex and one vertex of degree 4 are the only (nonadjacent) nonprobes. The enhanced graph is the 3-sun itself, which is not strongly chordal. A strongly chordal embedding is obtained by adding the edge between the two nonprobes.

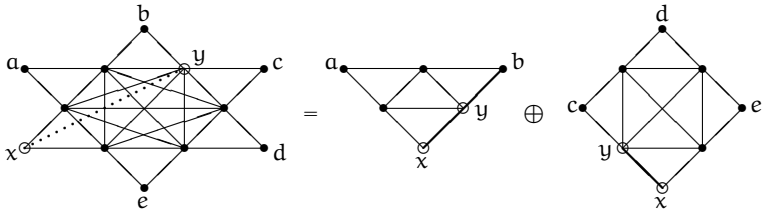


Fig. 5.3. The split of a 6-sun into a 3-sun and a 4-sun.

Lemma 5.37. *A sun can be embedded into a strongly chordal graph if and only if the underlying cycle has no* P_3 *of probes and the clique and independent set both have at least one nonprobe.*

Proof. Let S be a sun. Obviously, S must contain at least two nonadjacent nonprobes. Notice that, if an edge is added between two simplicials, this creates a C_4. Hence in this case there must also be an edge added between a simplicial and a vertex of the clique. This shows that there must be at least one nonadjacent pair of nonprobes in the clique and independent set. We prove by induction that there is no embedding when the cycle contains a P_3 of probes. This is easy to check for the case where S is a 3-sun. Assume S is a k-sun

for some $k > 3$. Consider an edge (x, y) added between a nonprobe of the clique and a nonprobe of the independent set. The edge (x, y) splits S into two smaller suns, or into a diamond[6] and a smaller sun. (See Figure 5.3.) The P_3 is contained in a smaller sun: Notice that it cannot be in a diamond, since any diamond that splits off has at least one simplicial nonprobe. By induction, there is no embedding of the smaller sun.
We prove the converse by induction. We show that there exists an embedding which adds edges only between vertices of the independent set and the clique. When S is a 3-sun, an embedding is obtained by adding an edge between a nonadjacent pair of nonprobes. Let S be a k-sun for some $k > 3$. We first show that the cycle must have a P_4 with nonprobe endpoints. This is clear when there is an edge with probe endvertices since there is no P_3 of probes. If there is no such edge, the claim follows since the probes and nonprobes cannot alternate. Let (x, y) be the middle edge of the P_4. Of the two, let x be the simplicial. Let a and b be the nonprobe neighbors in the cycle of x and y respectively. Add the edge (a, b). This splits S into a diamond and a $k-1$-sun S'. By induction, since S' has at least one pair of nonprobes a and b in the independent set and clique, and since the cycle of S' has no P_3 of probes, there exists an embedding of S'. Notice that $N[y] \subseteq N[a]$ in the embedding. That is, x is simple. This process eliminates the simplicial vertices one by one. In the end only the clique and one simplicial vertex remains. Obviously, this does not have a sun. This shows that there exists a simple elimination ordering, that is, the embedding is strongly chordal. □

We end this section by mentioning some conjectures that we have not been able to settle.

Conjecture 5.38. A partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$ has an embedding into a strongly chordal graph if and only if the enhanced graph G^* is chordal and every even cycle in G^* of length at least 6 either has an odd chord or two nonprobes at odd distance in the cycle.

We call a vertex *probe simple* if it can be made into a simple vertex by adding edges between nonprobes.

Conjecture 5.39. A graph is partitioned probe strongly chordal if and only if the enhanced graph is chordal and every induced subgraph has a probe simple vertex.

[6] A *diamond* is a 4-cycle with exactly one chord.

6

Partitioned Probe Comparability Graphs

In this chapter we show that there exist polynomial time algorithms for the recognition of partitioned probe comparability and cocomparability graphs. We give an $O(nm)$-time algorithm for the recognition of partitioned probe comparability graphs, where n and m are the numbers of vertices and edges, respectively.

By definition, a graph is a cocomparability graph if its complement is a comparability graph. Hence a partitioned graph G is a partitioned probe cocomparability graph if G^*, the sandwich conjugate of G, is a partitioned probe comparability graph. Thus partitioned probe cocomparability graphs can be recognized in $O(n^3)$ time by using the recognition algorithm for partitioned probe comparability graph.

A graph is a permutation graph if and only if it is at the same time a comparability graph and a cocomparability graph [170]. An immediate consequence is that a probe permutation graph is both probe comparability and probe cocomparability. For partitioned graphs, surprisingly, also the 'converse' remains true: G is a partitioned probe permutation graph if and only if G and G^* are both partitioned probe comparability graphs. This result leads to an $O(n^3)$ time recognition algorithm for partitioned probe permutation graphs.

6.1 Preliminaries

Unless stated otherwise, a graph is regarded as undirected. If the graph is directed we use the notation $\overrightarrow{xy}$ to denote the *arc* directed from x to y. Likewise, we use $\overleftarrow{xy}$ to denote the arc directed in the opposite direction. For a subset A of edges or arcs of a graph we denote by $V(A)$ the set of endvertices incident with elements of A.

We say that a vertex x and a set $U \subset V$ with $x \notin U$ of vertices are *completely adjacent*, *partially adjacent*, or *nonadjacent* if x is adjacent to every vertex of

U, if there exist $y, z \in U$ such that $(x, y) \in E$ and $(x, z) \notin E$, or if x is not adjacent to any vertex of U, respectively. If X and Y are disjoint subsets of V, and (x, y) is an edge of G with $x \in X$ and $y \in Y$, we say that (x, y) is an XY-*edge*. In case $X = \{x\}$, it is an xY-edge. Moreover, we say that X and Y are adjacent. If *every* pair (x, y) with $x \in X$ and $y \in Y$ is an edge of G, we say that X, Y are *completely adjacent*.

If X is a subset of the vertices of a partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$, then we write $\mathbb{P}(X) = X \cap \mathbb{P}$ and $\mathbb{N}(X) = X \cap \mathbb{N}$.

For a subset $\mathcal{E} \subseteq E$ of edges of a graph $G = (V, E)$, let $\overrightarrow{\mathcal{E}} = \{\overrightarrow{xy}, \overrightarrow{yx} \mid (x, y) \in \mathcal{E}\}$. If $\mathcal{E} = E$ we use $\overrightarrow{E} = \overrightarrow{\mathcal{E}}$. We call elements of $\overrightarrow{E}$ *directed edges* of G. For a set F of directed edges, we write $\widehat{F} = \{(x, y) \mid \overrightarrow{xy} \in F \text{ or } \overrightarrow{yx} \in F\}$ for its *symmetric closure*.

Definition 6.1. *Let $F \subseteq \overrightarrow{E}$ be a set of directed edges of a graph $G = (V, E)$. Define:*

1. $F^{-1} = \{\overleftarrow{xy} \mid \overrightarrow{xy} \in F\}$, *and*
2. $F^2 = \{\overrightarrow{xz} \mid \overrightarrow{xy} \text{ and } \overrightarrow{yz} \in F \text{ for some } y \in V\}$.

Definition 6.2. *Let $\mathcal{E} \subseteq E$ be a subset of edges of a graph $G = (V, E)$. We call F an* orientation *of $\mathcal{E}$ if $F + F^{-1} = \overrightarrow{\mathcal{E}}$ and $F \cap F^{-1} = \varnothing$.*

If $\mathcal{E} = E$, we call F also an orientation of G.

Definition 6.3. *A* transitive orientation *of a graph $G = (V, E)$ is an orientation F of E such that $F^2 \subseteq F$. A graph G is a* comparability graph *if G has a transitive orientation.*

Given a comparability graph, a transitive orientation of its edges can be obtained in linear time [99, 146].[1] However, checking the transitivity of the orientation needs a verification phase, for which no faster algorithm is known than, *e.g.*, a fast matrix multiplication (see, *e.g.*, [48, Chapter 26]).

Definition 6.4 (Golumbic [86]). *Define the binary relation Γ on the directed edges $\overrightarrow{E}$ of a graph $G = (V, E)$ as follows. For $(x, y),\ (x, z) \in E$,*

$$\overrightarrow{xy}\ \Gamma\ \overrightarrow{xz} \iff \overleftarrow{xy}\ \Gamma\ \overleftarrow{xz} \iff (y, z) \notin E$$

The relation Γ is reflexive and symmetric and its transitive closure Γ^c is an equivalence relation on $\overrightarrow{E}$. The equivalence classes of Γ^c partition $\overrightarrow{E}$ into the *implication classes* of G. For an implication class A of G, the *symmetric closure* of A, *i.e.*, $\widehat{A}$, is called a *color class* of G.

[1] For a bit more accessible algorithm, we refer to [62]. Notice however, that this algorithm runs in $O(n + m\alpha(n, m))$ time.

Golumbic [83, 84] gave a simple algorithm to test whether a graph $G = (V, E)$ is a comparability graph and to give it a transitive orientation if it is a comparability graph. The central part of Golumbic's algorithm is to compute a G-*decomposition* of $\overrightarrow{E}$, defined as follows.

Definition 6.5. *Let* $G = (V, E)$ *be a graph. A* G-decomposition *is a partition* $E = \hat{B}_1 + \cdots + \hat{B}_k$*, where* B_i *is an implication class of* $G\left[\hat{B}_i + \cdots + \hat{B}_k\right]$ *for* $i = 1, \ldots, k$.

Golumbic's algorithm follows directly from the following theorem.

Theorem 6.6 (TRO Theorem [83, 84, 86]). *Let* $G = (V, E)$ *be a graph and let* $E = \hat{B}_1 + \cdots + \hat{B}_k$ *be a* G*-decomposition. The following statements are equivalent.*

(i) G *is a comparability graph;*
(ii) $A \cap A^{-1} = \varnothing$ *for all implication classes* A *of* G*;*
(iii) $B_i \cap B_i^{-1} = \varnothing$ *for* $i = 1, \ldots, k$.

Furthermore, if these conditions hold, then $B_1 + \cdots + B_k$ *is a transitive orientation of* E.

By Theorem 6.6, we can test whether a graph $G = (V, E)$ is a comparability graph and give G a transitive orientation through computing a G-decomposition. Golumbic [83, 84, 86] gave an algorithm to compute a G-decomposition in $O(\Delta \cdot m)$ time where Δ is the maximum degree of a vertex in G.

6.2 Partitioned probe comparability graphs

In this section we extend the algorithm for recognizing comparability graphs given by Golumbic [83, 84, 86] to allow recognizing partitioned probe comparability graphs within the same time bound. This algorithm shows that a graph G is a comparability graph by showing that G has a transitive orientation. An orientation of a partitioned probe comparability graph may not be transitive, but the transitive completion may be a transitive orientation of an embedding. The following proposition is clear from the definitions.

Proposition 6.7. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned probe comparability graph with an embedding* H*. Let* $\mathcal{F}$ *be a transitive orientation of* H*,* $F = \overrightarrow{E} \cap \mathcal{F}$*, and let* (V, F^c) *be the transitive closure of* (V, F)*. Then,*

(i) $F^c \subseteq \mathcal{F}$*,*
(ii) $V(\mathcal{F} - F) \subseteq \mathbb{N}$*, and*
(iii) $(V, \hat{F}^c)$ *is a comparability graph with transitive orientation* F^c.

Observe that F^c is an orientation of the smallest embedding of G such that it can be oriented in agreement with F. We will call F a quasitransitive orientation. Determining whether G is partitioned probe comparability will be equivalent to determining whether it has a quasitransitive orientation.

Definition 6.8. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph and let* F *be an orientation of* G. *We call* F quasitransitive *if* $V(F^2 - F) \subseteq \mathbb{N}$.

Theorem 6.9. *A partitioned graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *is partitioned probe comparability if and only if* G *has a quasitransitive orientation.*

Proof. Using the notation of Proposition 6.7, if G is partitioned probe comparability with embedding H, then F is clearly a quasitransitive orientation of G. That is, if $\overrightarrow{xy}, \overrightarrow{yz} \in F \subseteq \mathcal{F}$, then $\overrightarrow{xz} \in \mathcal{F}$ and, unless x and z are both nonprobes, $\overrightarrow{xz} \in F$.
Now suppose F is a quasitransitive orientation of G. Let $\mathcal{F} = F + F^2$. We prove that G is a partitioned probe comparability graph by showing that $(V, \mathcal{F})$ is transitive, *i.e.*, by showing that $\mathcal{F}^2 \subseteq \mathcal{F}$. Suppose both $\overrightarrow{xy}, \overrightarrow{yz} \in \mathcal{F}$. We show that $\overrightarrow{xz} \in \mathcal{F}$.
If y is a probe, then $\overrightarrow{xy}, \overrightarrow{yz} \in F$ and hence $\overrightarrow{xz} \in F^2 \subseteq \mathcal{F}$.
Suppose instead that y is a nonprobe. If x is also a nonprobe, then $\overrightarrow{xy} \notin F$. Therefore $\overrightarrow{xy} \in F^2$ and there exists a $u \in \mathbb{P}$ such that $\overrightarrow{xu}, \overrightarrow{uy} \in F$. Similarly, if $z \in \mathbb{N}$, there exists a $v \in \mathbb{P}$ such that $\overrightarrow{yv}, \overrightarrow{vz} \in F$. Notice that $u \neq v$; otherwise the edge (u, y) would be oriented in two directions. Thus if x, y, and z are all nonprobes, we obtain

$$\begin{aligned} v \in \mathbb{P} \text{ and } \overrightarrow{uy},\ \overrightarrow{yv} \in F &\implies \overrightarrow{uv} \in F \\ v \in \mathbb{P} \text{ and } \overrightarrow{xu},\ \overrightarrow{uv} \in F &\implies \overrightarrow{xv} \in F \\ \overrightarrow{xv},\ \overrightarrow{vz} \in F &\implies \overrightarrow{xz} \in F^2 \subseteq \mathcal{F}. \end{aligned}$$

If x and y are nonprobes but z is a probe, then $\overrightarrow{uy}, \overrightarrow{yz} \in F$ implies that $\overrightarrow{uz} \in F$ and again $\overrightarrow{xz} \in F^2$. The case that x is a probe but z is a nonprobe is similar. Thus G is a probe comparability graph and $H = (V, \hat{\mathcal{F}})$ is an embedding of G. □

We need to modify the Γ relation to use it in partitioned probe comparability recognition.

Definition 6.10. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph. We define a binary relation* Υ *on* $\overrightarrow{E}$ *as follows. Let* $(x, y), (x, z) \in E$ *be edges of* G. *Then each of* $\overrightarrow{xy}\ \Upsilon\ \overrightarrow{xz}$ *and* $\overrightarrow{yx}\ \Upsilon\ \overrightarrow{zx}$ *if and only if one of the following holds:*

(a) $y = z$, *or*
(b) $(y, z) \notin E$ *and at least one of* y *and* z *is a probe.*

The relation Υ is reflexive and symmetric. Its transitive closure, denoted by Υ^c, defines an equivalence relation on $\overrightarrow{E}$. We call the equivalence classes the *probe implication classes* of G. Let A be a probe implication class of G. We call the symmetric closure of A, *i.e.*, $\widehat{A}$, a *probe color class* of G. We define the *probe* G*-decomposition* as follows:

Definition 6.11. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph. A partition* $E = \widehat{B}_1 + \cdots + \widehat{B}_k$ *is called a* probe G-decomposition *if* B_i *is a probe implication class of* $G\left[\widehat{B}_i + \cdots + \widehat{B}_k\right]$ *for each* $1 \leq i \leq k$.

The following extension of Theorem 6.6 is the basis for our algorithm to recognize and orient partitioned probe comparability graphs.

Theorem 6.12 (Probe TRO Theorem). *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph with probe* G*-decomposition* $E = \widehat{B}_1 + \cdots + \widehat{B}_k$. *The following statements are equivalent.*

(i) G *is a partitioned probe comparability graph;*
(ii) $A \cap A^{-1} = \varnothing$ *for all probe implication classes* A *of* G*;*
(iii) $B_i \cap B_i^{-1} = \varnothing$ *for* $i = 1, \ldots, k$.

Furthermore, if these conditions hold, then $F = B_1 + \cdots + B_k$ *is a quasitransitive orientation of* E *and* $H = (V, \widehat{\mathcal{F}})$ *is a comparability graph which is an embedding of* G*, where* $\mathcal{F} = F + F^2$.

Theorem 6.12 extends the recognition algorithm for comparability graphs given in [86] to a recognition algorithm for partitioned probe comparability graphs and assigning a quasitransitive orientation. An embedding follows from the quasitransitive orientation. We postpone the proof of the Probe TRO Theorem. Some of the following lemmas are extensions of lemmas given in [83, 84, 86, 178] for proving the TRO Theorem.

Arcs $\overrightarrow{xy}$ and $\overrightarrow{uv}$ are in the same probe implication class if and only if they are joined by an Υ*-chain*, *i.e.*, a sequence of edges $(x_i, y_i) \in E$ such that

$$\overrightarrow{xy} = \overrightarrow{x_0y_0} \,\Upsilon\, \cdots \,\Upsilon\, \overrightarrow{x_ky_k} = \overrightarrow{uv}. \tag{6.1}$$

Lemma 6.13. *If* $\overrightarrow{xy}\ \Upsilon^c\ \overrightarrow{uv}$*, then there exists an* Υ*-chain* (6.1) *such that for each* i, $1 \leq i \leq k$, *either*

$$\begin{aligned} &x_{i-1} = x_i \text{ and } y_{i-1} \neq y_i, \text{ or} \\ &x_{i-1} \neq x_i \text{ and } y_{i-1} = y_i. \end{aligned} \tag{6.2}$$

Such a chain will be called a canonical Υ-chain.

Proof. Suppose $\overrightarrow{xy}\ \Upsilon^c\ \overrightarrow{uv}$ and that (6.1) is a shortest Υ-chain from $\overrightarrow{xy}$ to $\overrightarrow{uv}$. Then (6.2) holds for each $1 \leq i \leq k$, since otherwise the i^{th} step is equality and we have a shorter chain by removing it. □

Corollary 6.14. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph and let* A *be a probe implication class of* G*. Then* $(V(A), \hat{A})$ *is a connected graph.*

Lemma 6.15 (The Probe Triangle Lemma). *Let* x*,* y*, and* z *be three distinct vertices of a partitioned graph* $G = (\mathbb{P} + \mathbb{N}, E)$*, let* $z \in \mathbb{P}$*, and let* X*,* Y*, and* Z *be probe implication classes of* G *with* $X \neq Z$ *and* $Z \neq Y^{-1}$ *and having arcs* $\overrightarrow{xy} \in Z$*,* $\overrightarrow{zx} \in Y$*, such that* $\overrightarrow{zy} \in X$*. Then the following four statements hold.*

1. *If* $x \neq u \neq y$*,* $(x, u) \in E$ *and* $\overrightarrow{xu} \Upsilon \overrightarrow{xy}$*, then* $z \neq u$*,* $(z, u) \in E$ *and* $\overrightarrow{zu} \in X$*.*
2. *If* $x \neq u \neq y$*,* $(u, y) \in E$ *and* $\overrightarrow{uy} \Upsilon \overrightarrow{xy}$*, then* $z \neq u$*,* $(z, u) \in E$ *and* $\overrightarrow{zu} \in Y$*.*
3. *If* $\overrightarrow{pq} \in Z$*, then* $\overrightarrow{zp} \in Y$ *and* $\overrightarrow{zq} \in X$*.*
4. $z \notin V(Z)$*.*

Proof. We first prove 1. Notice that $\overrightarrow{xz} \notin Z$, since $\overrightarrow{zx} \in Y$ and $Z \neq Y^{-1}$. Then $u \neq z$ because $\overrightarrow{xy} \in Z \neq Y^{-1}$ and $\overrightarrow{xu} \Upsilon \overrightarrow{xy}$. Since $z \in \mathbb{P}$, if $(z, u) \notin E$, $\overrightarrow{xz} \Upsilon \overrightarrow{xu}$, a contradiction to the assumption that $\overrightarrow{xz} \notin Z$. Hence $(z, u) \in E$ must hold. Since $\overrightarrow{xu} \Upsilon \overrightarrow{xy}$, we have that $(y, u) \notin E$ and at least one of y and u is a probe. Because $(z, u) \in E$, $(z, y) \in E$, $(y, u) \notin E$, and at least one of y and u is a probe, we have $\overrightarrow{zu} \Upsilon \overrightarrow{zy}$ and $\overrightarrow{zu} \in X$ because $\overrightarrow{zy} \in X$.
The proof of 2 is similar.
Next, to prove 3, let $\overrightarrow{pq} \in Z$. By Lemma 6.13, since $\overrightarrow{xy} \in Z$, there exists a canonical Υ-chain

$$\overrightarrow{xy} = \overrightarrow{x_0y_0} \Upsilon \cdots \Upsilon \overrightarrow{x_ky_k} = \overrightarrow{pq}.$$

We claim that $x_i \neq z \neq y_i$, $\overrightarrow{zx_i} \in Y$, and $\overrightarrow{zy_i} \in X$, for $0 \leq i \leq k$. We prove the claim by induction on i. It holds for $i = 0$ by assumption. Suppose it holds for $i - 1$. If $x_{i-1} = x_i$ and $y_i \neq y_{i-1}$, then $\overrightarrow{zy_i} \in X$ by 1. Otherwise $x_{i-1} \neq x_i$ and $y_i = y_{i-1}$ and hence $\overrightarrow{zx_i} \in Y$ by 2. In either case, we have $x_i \neq z \neq y_i$, $\overrightarrow{zx_i} \in Y$ and $\overrightarrow{zy_i} \in X$. In particular, $p = x_k \neq z \neq y_k = q$, $\overrightarrow{zp} = \overrightarrow{zx_k} \in Y$, and $\overrightarrow{zq} = \overrightarrow{zy_k} \in X$. Thus 3 and 4. □

The following is immediate from (4) of Lemma 6.15.

Corollary 6.16. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph with a probe implication class* Z*. Let* $x, y, z \in V = \mathbb{P} + \mathbb{N}$*, let* $(x, y), (x, z), (y, z) \in E$*, and let* $z \in \mathbb{P}$*. If* $\overrightarrow{xy} \in Z$*, then* $z \in V(Z)$ *if and only if at least one of* $\overrightarrow{xz} \in Z$ *or* $\overrightarrow{zy} \in Z$*.*

Lemma 6.17. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph and let* A *be a probe implication class of* G*. Exactly one of the following alternatives holds.*

(i) *either* $A = A^{-1}$ *or,*
(ii) $A \cap A^{-1} = \varnothing$*, and* A *and* A^{-1} *are quasitransitive orientations of the graph* $G_A = (V(G), \hat{A})$*.*

Proof. If $A \cap A^{-1} \neq \varnothing$, then there exists an arc $\overrightarrow{xy} \in A \cap A^{-1}$, and $\overrightarrow{xy}\ \Upsilon^c\ \overrightarrow{yx}$. For any $\overrightarrow{uv} \in A$, we have $\overrightarrow{uv}\ \Upsilon^c\ \overrightarrow{xy}$ and $\overrightarrow{yx}\ \Upsilon^c\ \overrightarrow{vu}$, implying $\overrightarrow{uv}\ \Upsilon^c\ \overrightarrow{vu}$ and $\overrightarrow{vu} \in A$. Thus $A = A^{-1}$.
On the other hand suppose $A \cap A^{-1} = \varnothing$. We will show that $V(A^2 - A) \subseteq \mathbb{N}$ by showing that for $\overrightarrow{xy}, \overrightarrow{yz} \in A$ where at least one of x and z is a probe of G, $\overrightarrow{xz} \in A$. If $(x,z) \notin E$, then $\overrightarrow{xy}\ \Upsilon\ \overrightarrow{zy}$ and $\overrightarrow{zy} \in A \cap A^{-1} \neq \varnothing$, a contradiction. Thus $(x,z) \in E$ must hold. If $\overrightarrow{xz} \in A$, then we are done. Suppose $\overrightarrow{xz} \notin A$. Consider the case of $x \in \mathbb{P}$. We have $\overrightarrow{yz} \in A$ and $\overrightarrow{yx}, \overrightarrow{xz} \notin A$. By Corollary 6.16, $x \notin V(A)$, a contradiction. The case of $z \in \mathbb{P}$ is similar. Thus A is a quasitransitive orientation of G_A.
Obviously, quasitransitivity of A implies quasitransitivity of A^{-1} for G_A. □

Lemma 6.18. *Let* $G = (V, E) = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph, and let* A *be a probe implication class of* G. *If* F *is a quasitransitive orientation of* $G_{\bar{A}} = (V, E - \hat{A})$ *and if* $A \cap A^{-1} = \varnothing$, *then* $F + A$ *is a quasitransitive orientation of* G.

Proof. By the previous lemma, A is a quasitransitive orientation of $G_A = (V, \hat{A})$. Let $\mathcal{F} = F + A$. Clearly $\mathcal{F}$ is an orientation of G and $F \cap A = \varnothing$. If $\mathcal{F}$ is not quasitransitive, then there exist arcs $\overrightarrow{xy}, \overrightarrow{yz} \in \mathcal{F}$, x and z not both nonprobes, such that $\overrightarrow{xz} \notin \mathcal{F}$. If $(x,z) \notin E$ then $\overrightarrow{xy}\ \Upsilon\ \overrightarrow{zy}$, contradicting quasitransitivity of A unless both $\overrightarrow{xy}, \overrightarrow{zy} \in F$. But then $\overrightarrow{xy}$ and $\overrightarrow{yz}$ violate quasitransitivity of $G_{\bar{A}}$. Suppose then that $\overrightarrow{zx} \in \mathcal{F}$. Two of the three arcs, $\overrightarrow{xy}, \overrightarrow{yz}$, and $\overrightarrow{zx}$, must be in A, or in F. We have a violation of quasitransitivity in G_A, or in $G_{\bar{A}}$, respectively. □

Notice that for every implication class A, $V(A)$ is a module in G [86, pp. 112]. We define the following generalization which includes probe implication classes.

Definition 6.19. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph, and let* $M \subseteq V = \mathbb{P} + \mathbb{N}$. *Let* $C_1, \ldots, C_t$ *be the components of* $G[\mathbb{P}(M)]$. *We call* M *a* QT-module *if the following conditions are satisfied.*
(a) $\forall_{x \in \mathbb{P} - M}$ *either* $N(x) \cap M = \varnothing$ *or* $M \subseteq N(x)$.
(b) $\forall_{y \in \mathbb{N} - M} \forall_{1 \leq i \leq t}$ *either* $N(y) \cap V(C_i) = \varnothing$ *or* $V(C_i) \subseteq N(y)$.

Lemma 6.20. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph and let* A *be a probe implication class of* G. *Then* $V(A)$ *is a* QT*-module.*

Proof. First suppose there exists an $x \in \mathbb{P} - V(A)$ which is connected to some, but not all, vertices in $V(A)$. Define $R = \{w \in V(A) \mid (x,w) \in E\}$ and $U = V(A) - R$. By Corollary 6.14, the undirected graph $(V(A), \hat{A})$ is connected. There exist $p \in U$ and $q \in R$ such that $\overrightarrow{pq} \in A$ or $\overleftarrow{pq} \in A$. Since $(p,q), (q,x) \in E$, $(p,x) \notin E$, and $x \in \mathbb{P}$, we obtain

$$\overrightarrow{pq} \Upsilon \overrightarrow{xq} \quad \text{and} \quad \overleftarrow{pq} \Upsilon \overleftarrow{xq}, \tag{6.3}$$

contradicting $x \notin V(A)$. We have shown that condition (a) of Definition 6.19 is satisfied.
Instead suppose there exist a $y \in \mathbb{N} - V(A)$ and a component C_i of $\mathbb{P}(V(A))$ such that y is adjacent to some but not all vertices in C_i. The proof is the same as before, except that now R and U are the neighbors and nonneighbors of y restricted to C_i, and that this time p is a probe in (6.3) (instead of x). Therefore condition (b) of Definition 6.19 is also satisfied. □

Lemma 6.21. *Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned graph, let M be a QT-module and let A be a probe implication class of G. Then either*

$$V(A) \subseteq M \quad \text{or} \quad A \cap (M \times M) = \varnothing.$$

Proof. Suppose that $\overrightarrow{xy} \Upsilon \overrightarrow{zy}$ for some edges $(x, y), (y, z) \in E$ with $x, y \in M$ and $z \in V - M$. If $z \in \mathbb{P}$, Definition 6.19 gives us that $(x, z) \in E$. If $z \in \mathbb{N}$ and $(x, z) \notin E$, then x cannot be a probe, because otherwise x and y are in the same component of $\mathbb{P}(M)$. Either case contradicts $\overrightarrow{xy} \Upsilon \overrightarrow{zy}$, from which the lemma follows. □

We are now ready to prove our main theorem.

Theorem 6.22. *Let $G = (V = \mathbb{P} + \mathbb{N}, E)$ be a partitioned graph. The following statements are equivalent.*

(i) *G is a partitioned probe comparability graph.*
(ii) *$A \cap A^{-1} = \varnothing$ for each probe implication class A of G.*
(iii) *For* each *probe implication class A of G, the graphs $G_A = (V, \hat{A})$ and $G_{\bar{A}} = (V, E - \hat{A})$ are partitioned probe comparability graphs.*
(iv) *For* some *probe implication class A of G, the graphs $G_A = (V, \hat{A})$ and $G_{\bar{A}} = (V, E - \hat{A})$ are partitioned probe comparability graphs.*

Proof. We will prove the theorem by induction on the number of vertices in G, and on the number k of probe color classes when two graphs have the same number of vertices.
Suppose $k = 1$. Since G has only one probe color class, $E(G_{\bar{A}}) = \varnothing$, and by Lemma 6.17, all the statements hold or none hold.
For the remainder of the proof we assume that $k > 1$, and that the theorem holds for all partitioned probe graphs on fewer vertices than G, and for all partitioned probe graphs on $V(G)$ for which the number of probe color classes is less than k. We prove that the statements are equivalent for a partitioned probe graph G of k probe color classes.
(i)⇒(ii). Let F be a quasitransitive orientation of G and suppose that $\overrightarrow{xy} \in A \cap A^{-1}$. Then there is an Υ-chain from $\overrightarrow{xy}$ to $\overrightarrow{yx}$

$$\overrightarrow{xy} = \overrightarrow{x_1y_1} \Upsilon \cdots \Upsilon \overrightarrow{x_\ell y_\ell} = \overrightarrow{yx}.$$

However, from the definitions of Υ and quasitransitive orientation for G, if (x, y) and (p, q) are two edges of G such that $\overrightarrow{xy} \in F$ and $\overrightarrow{xy} \Upsilon \overrightarrow{pq}$ then $\overrightarrow{pq} \in F$, which is a contradiction since $\overrightarrow{xy}$ and $\overrightarrow{yx}$ cannot both be in F.

(ii)⇒(iii). That G_A is a partitioned probe comparability graph follows from (ii) of Lemma 6.17. In the following we prove that $G_{\bar{A}}$ is also a partitioned probe comparability graph. We consider two cases:

CASE 1. $V(A) = V$. First we show that every probe implication class D of G, with $\hat{D} \neq \hat{A}$, is a subset of some probe implication class of $G_{\bar{A}}$. Let $\Upsilon_{\bar{A}}$ denote the relation Υ for the graph $G_{\bar{A}}$. Suppose $\overrightarrow{xy}, \overrightarrow{xz} \in D$ and $\overrightarrow{xy} \Upsilon \overrightarrow{xz}$. Then $(x, z) \notin E$ and clearly $\overrightarrow{xy} \Upsilon_{\bar{A}} \overrightarrow{xz}$. That is, an Υ-chain between two arcs of D implies an $\Upsilon_{\bar{A}}$-chain between them. Thus D is a subset of some probe implication class of $G_{\bar{A}}$.

We will show that probe implication classes of G that merge in $G_{\bar{A}}$ are all stars in $G_{\bar{A}}$ with a nonprobe as the center, and any two stars which merge together have a common center as the source for all arcs or the sink for all arcs in the classes. Therefore $D \cap D^{-1} = \varnothing$ for every probe implication class D of $G_{\bar{A}}$. Since $G_{\bar{A}}$ has at least one class fewer than G, the result now follows from the induction hypothesis.

Suppose the relation $\Upsilon_{\bar{A}}$ connects $\overrightarrow{xy} \in D_i$ and $\overrightarrow{xz} \in D_j$, where D_i and D_j are two distinct probe implication classes of G. Then $(y, z) \in \hat{A}$. If $x \in \mathbb{P}$, then by Lemma 6.15 (4) $x \notin V(A)$, contradicting the assumption $V(A) = V(G)$; thus $x \in \mathbb{N}$, and y and z must be probes. Also note that $D_i \neq D_j^{-1}$, since otherwise $\overrightarrow{zx}, \overrightarrow{xy} \in D_i$, but $\overrightarrow{zy} \notin D_i$. Therefore $\hat{D}_i \cap \hat{D}_j = \varnothing$.

Suppose there exists a vertex $p \neq x$ such that $\overrightarrow{xy} \Upsilon \overrightarrow{py}$. Then $p \in \mathbb{P}$ since $x \in \mathbb{N}$. Since $z \in \mathbb{P}$ and $\overrightarrow{xy}, \overrightarrow{py} \in D_i$, we have $\overrightarrow{pz} \in D_j$ by the Probe Triangle Lemma (iii). Then by (iv) of the same lemma, since $\overrightarrow{py} \in D_i$, $\overrightarrow{pz} \in D_j$, and $\overrightarrow{yz} \in \hat{A}$, we have $p \notin V(A)$, a contradiction. Hence all arcs of the probe implication class of $\overrightarrow{xy}$ must be of the form $\overrightarrow{xq}$. Hence D_i is a star, as claimed. Since $\overrightarrow{xy} \in D_i$ and $\overrightarrow{xz} \in D_j$, we see that D_i and D_j merge at x.

CASE 2. $V(A) \subset V$. As in Case 1, every probe implication class of G except A and A^{-1} is contained in some probe implication class of $G_{\bar{A}}$. Therefore $G_{\bar{A}}$ has at least one color class fewer than G. The result follows from the induction hypothesis if $D \cap D^{-1} = \varnothing$ for every implication class of $G_{\bar{A}}$.

We divide the arcs of $\overrightarrow{E}(G_{\bar{A}})$ into two groups: those arcs between two vertices of $M = V(A)$, and those with at least one endvertex not in M. Since M is a QT-module, by Lemma 6.21, every probe implication class of G is either a subset of $\overrightarrow{E}(G[M])$, or disjoint from $\overrightarrow{E}(G[M])$. First consider $G[M]$. Relation Υ does not connect any arc of $\overrightarrow{E}(G[M])$ with any arc not in $\overrightarrow{E}(G[M])$, and $\Upsilon_{\bar{A}}$ does not either, because the edges of G missing in $G_{\bar{A}}$ do not leave M.

Thus, by the same argument as in Case 1, every probe implication class D of $(M, E(G[M]) - A)$ satisfies $D \cap D^{-1} = \varnothing$.
Next we consider those probe implication classes that are disjoint from $\overrightarrow{E}(G[M])$. Let X be a set containing one vertex from each component of $G[\mathbb{P}(M)]$. Since $G[V - M + X]$ is a proper induced subgraph, it satisfies statement (ii) of the theorem. Since the vertex set is not all of G, the induction hypothesis says that $G[V - M + X]$ is a partitioned probe comparability graph having a quasitransitive orientation F. We extend this orientation to an orientation F^* of $(V, E - E(G[M]))$ as follows. Let $v \in V - M$. Since X is an independent set, every arc between v and X in F is from v to X, or every such arc is from X to v. We give every edge between v and M the same orientation.
We show that F^* is again a quasitransitive orientation. Suppose there is a violation of quasitransitivity in F^* involving $(v, x), (v, y) \in E$. If $v \in V - M$, and $x, y \in M$, by the construction of F^*,

$$\overrightarrow{vx} \in F^* \iff \overrightarrow{vy} \in F^*$$

and there is no violation.
Next, assume $x \in M$ but $v, y \in V - M$, and x and y not both nonprobes. If $x \in \mathbb{P}$, let $z \in X$ be the vertex, possibly the same as x, in the same component of $\mathbb{P}(G[M])$ as x; otherwise, if $x \in \mathbb{N}$, let $z \in X$ be arbitrary. Then

$$(z, y) \in E \iff (x, y) \in E$$

because M is a QT-module. Since (v, x) and (v, z) have the same direction, as well as (x, y) and (z, y) if they are both edges, and since $\{z, y, v\}$ does not contain a violation of quasitransitivity, neither does $\{x, y, v\}$.
Suppose now $v \in M$ and $x, y \in V - M$. Then there exists $z \in X$ such that (z, x), $(z, y) \in E$ and such that (z, x) and (z, y) receive the same orientations as (v, x) and (v, y), respectively. Since F is quasitransitive, $\{x, y, z\}$ does not contain a violation of quasitransitivity; neither therefore does $\{x, y, v\}$.
We conclude that $(V, E - E(G[M]))$ is a partitioned probe comparability graph with quasitransitive orientation F^* and with fewer color classes than G. By the inductive hypothesis $D \cap D^{-1} = \varnothing$ for every implication class D of this graph. Thus we have $D \cap D^{-1} = \varnothing$ for every probe implication class D of $G_{\bar{A}}$, which by induction on the number of classes is a partitioned probe comparability graph.
(iii)⇒(iv) is clear.
(iv)⇒(i). This implication is precisely Lemma 6.18. □

Another characterization follows immediately.

Theorem 6.23. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph and let* A *be a probe implication class of* G. *Assume that* $V - V(A) \neq \varnothing$ *and that* $G[\mathbb{P}(V(A))]$ *has* ℓ

connected components $V_1, \ldots, V_\ell$. *Let* $X = \{x_i \mid 1 \leq i \leq \ell$ *and* $x_i \in \mathbb{P}(V_i)\}$. *Then* G *is a partitioned probe comparability graph if and only if* $G[V - V(A) + X]$ *and* $G[V(A)]$ *are both partitioned probe comparability.*

Proof. The class of probe comparability graphs is hereditary. Therefore, if G is in the class, then so are the two induced subgraphs. To get the reverse result, we use the same construction as in case 2, (ii)⇒(iii), of the proof of Theorem 6.22 to get $D \cap D^{-1} = \varnothing$ for every probe implication class of G, and thus G is a partitioned probe comparability graph. □

Also the Probe TRO Theorem follows immediately from Theorem 6.22.

We now present an algorithm for recognizing partitioned probe comparability graphs which is a modification of the algorithm given in Golumbic [86] for recognizing comparability graphs. The algorithm computes a probe G-decomposition, $\hat{B}_1, \ldots, \hat{B}_k$ for G. It uses the function

$$\mathsf{CLASS}(i,j) = \begin{cases} h & \text{if } \overrightarrow{v_i v_j} \text{ has been assigned to } B_h, \\ -h & \text{if } \overrightarrow{v_i v_j} \text{ has been assigned to } B_h^{-1}, \\ \text{undefined} & \text{if } \overrightarrow{v_i v_j} \text{ has not yet been assigned,} \end{cases}$$

and a variable FLAG which is 0 if $B \cap B^{-1} = \varnothing$ for each class B in the decomposition, and 1 otherwise. If the algorithm terminates with $\mathsf{FLAG} = 0$, then a quasitransitive orientation of G is obtained by combining all edges having positive CLASS.

The algorithm finds a probe color class $\hat{B}_h$ of the current graph, deletes it, and iterates. It calculates $\hat{B}_h$ by arbitrarily finding an arc $\overrightarrow{xy}$ which has not yet been assigned to any $\hat{B}_h$ for $1 \leq h < i$ and visiting all arcs $\overrightarrow{pq}$ with $\overrightarrow{xy}\ \Upsilon^c\ \overrightarrow{pq}$ using the DFS-like procedure $\mathsf{Quasi-EXPLORE}(i,j)$. The variable FLAG changes from 0 to 1 whenever $B_h \cap B_h^{-1} \neq \varnothing$. By Theorem 6.12 G is a partitioned probe comparability graph if and only if FLAG never changes. $\mathsf{Quasi-EXPLORE}(i,j)$ differs from $\mathsf{EXPLORE}(i,j)$ given by Golumbic in checking one more condition, $\{v_i, v_h\} \notin \mathbb{N}(G)$, when testing whether $\overrightarrow{v_i v_j}\ \Upsilon\ \overrightarrow{v_i v_h}$. The details of the algorithm and $\mathsf{Quasi-EXPLORE}(i,j)$ are given in Algorithm 1 and Algorithm 2, respectively.

It is easy to see that the modified algorithm can be implemented in the same time bound as the algorithm given by Golumbic. Hence we have the following theorem.

Theorem 6.24. *Recognition of partitioned probe comparability graphs and finding a quasitransitive orientation can be done in* $O(\Delta \cdot m)$ *time and* $O(n + m)$ *space, where* Δ *is the maximum degree of a vertex. Moreover, an embedding can also be obtained from a quasitransitive orientation in* $O(\Delta \cdot m)$ *time.*

Algorithm 1 Recognizing a partitioned probe comparability graph.

Input: The directed version $(V(G), \overrightarrow{E}(G))$ of a partitioned probe graph G with vertices $v_1, v_2, \ldots, v_n$ whose adjacency sets obey $j \in Adj(i)$ if and only if $\overrightarrow{v_i v_j} \in \overrightarrow{E}(G)$.
Output: A variable FLAG and a variable CLASS(i,j) for each edge $\overrightarrow{v_i v_j}$. If the algorithm terminates with FLAG equal to zero, then a quasitransitive orientation of G is obtained by combining all edges having positive CLASS.

```
1: Initialize: k ← 0; FLAG ← 0;
2: for each edge v_i v_j ∈ E(G) do
3:    if CLASS(i, j) is undefined then
4:       k ← k + 1;
5:       CLASS(i, j) ← k; CLASS(j, i) ← −k;
6:       Quasi-EXPLORE(i, j);
7:    end if
8: end for
```

Algorithm 2 procedure Quasi-EXPLORE(i,j)

Input: $\overrightarrow{v_i v_j}$.
Output: Find the edges of a probe implication class containing edge $\overrightarrow{v_i v_j}$.
For each edge with CLASS undefined, assign it a value.

```
1: for each h ∈ Adj(i) such that
   {v_h, v_j} ⊈ N(G) and h ∉ Adj(j) or |CLASS(j, h)| < k do
2:    if CLASS(i, h) is undefined then
3:       CLASS(i, h) ← k; CLASS(h, i) ← −k;
4:       Quasi-EXPLORE(i, h);
5:    else
6:       if CLASS(i, h) = −k then
7:          CLASS(i, h) = k; FLAG ← 1;
8:          Quasi-EXPLORE(i, h);
9:       end if
10:   end if
11: end for
12: for each h ∈ Adj(j) such that
   {v_h, v_i} ⊈ N(G) and h ∉ Adj(i) or |CLASS(i, h)| < k do
13:   if CLASS(h, j) is undefined then
14:      CLASS(h, j) ← k; CLASS(j, h) ← −k;
15:      Quasi-EXPLORE(h, j);
16:   else
17:      if CLASS(h, j) = −k then
18:         CLASS(h, j) = k; FLAG ← 1;
19:         Quasi-EXPLORE(h, j);
20:      end if
21:   end if
22: end for
```

Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned graph. Recall that the *sandwich conjugate* G^* of G is the partitioned graph obtained from $\overline{G}$ by removing all edges between vertices of $\mathbb{N}$. By the above definition, we have that a partitioned graph is a partitioned probe cocomparability graph if and only if its sandwich conjugate G^* is a partitioned probe comparability graph. Thus we have the following corollary.

Corollary 6.25. *The recognition of partitioned probe cocomparability graphs can be done in* $O(n^3)$ *time.*

6.3 A partitioned probe Dushnik & Miller

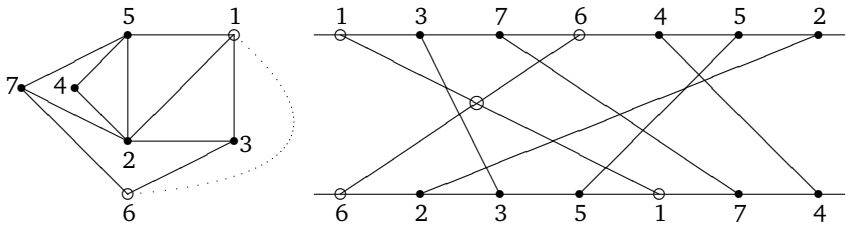


Fig. 6.1. A partitioned probe permutation graph and an intersection diagram of an embedding. Vertices 1 and 6 are nonprobes. The other vertices are probes.

Definition 6.26 ([70, 170]). *Let* $\pi \in \mathrm{Sym}(n)$ *be a permutation acting on the set of integers* $\{1, 2, \ldots, n\}$*. We define the* inversion graph $G[\pi]$ *as follows. The graph has vertex* $V = \{x_1, \ldots, x_n\}$ *and edge set* E *defined by*

$$(x_i, x_j) \in E \iff (i - j)\,(\pi^{-1}(i) - \pi^{-1}(j)) < 0.$$

An undirected graph G *is called a* permutation graph *if there exists a permutation* π *such that* $G \cong G[\pi]$.

Geometrically, we can diagram an inversion graph as follows. Write the integers $1, \ldots, n$ left to right as distinct points on a line, and write the permutation π of $\{1, \ldots, n\}$ on a parallel line. The vertices of $G[\pi]$ are the line segments joining points with the same label, and two vertices are joined by an edge if their line segments intersect.

Dushnik and Miller showed that a graph G is a permutation graph if and only if both G and $\overline{G}$ are comparability graphs [69]. This characterization permits permutation graphs to be recognized in linear time [146]. In this section we show that this characterization extends to partitioned probe permutation graphs where $\overline{G}$ is replaced by the sandwich conjugate G^*.

Lemma 6.27. *Let* $G = (\mathbb{P}+\mathbb{N}, E)$ *be a partitioned graph. Then* G *is a partitioned probe permutation graph if and only if there exists a labeling* L *of the vertices by integers* $1, \ldots, n$*, and a permutation* $\pi \in \mathrm{Sym}(n)$ *such that* $(x, y) \in E$ *if and only if both*

$$\{x, y\} \not\subseteq \mathbb{N} \quad \text{and} \quad (L(x) - L(y))\left(\pi^{-1}L(x) - \pi^{-1}L(y)\right) < 0. \tag{6.4}$$

Proof. It is easy to see that if $(x, y) \in E(G[\pi]) - E$ is an edge, then $\{x, y\} \subseteq \mathbb{N}$. Thus $G[\pi]$ is an embedding of G. □

Recall that the *sandwich conjugate* G^* of a partitioned probe graph G is the graph obtained from $\overline{G}$ by removing all edges between vertices of $\mathbb{N}$.

Theorem 6.28 (The Partitioned Probe Dushnik & Miller). *A partitioned probe graph* G *is a probe permutation graph if and only if both* G *and* G^* *are partitioned probe comparability graphs.*

Proof. Let G be a partitioned probe permutation graph and let H be an embedding of G. Both H and $\overline{H}$ are comparability graphs. By definition, $\{x, y\} \subseteq \mathbb{N}$ if either $(x, y) \in E(H) - E(G)$ or $(x, y) \in E(\overline{H}) - E(\overline{G})$. Thus both G and G^* are partitioned probe comparability graphs.
Suppose both G and G^* are partitioned probe comparability graphs. By Theorem 6.9, both G and G^* have quasitransitive orientations. Let F_1 and F_2 be quasitransitive orientations of G and G^*, respectively. We claim that $(V, F_1 + F_2)$ is an acyclic digraph. If not, let $v_0, \ldots, v_\ell, v_0$ be a cycle of the smallest possible length $\ell > 3$. Since at least one of the two ends of an edge is a probe, without loss of generality let $v_0 \in \mathbb{P}$. Then either $\overleftarrow{v_0v_2} \in F_1 + F_2$, in which case v_0, v_1, v_2, v_0 is a shorter cycle, or $\overrightarrow{v_0v_2} \in F_1 + F_2$, in which case $v_0, v_2, v_3, \ldots, v_0$ is a shorter cycle, contradicting minimality in each case.
If $\ell = 3$, then at least two of the three vertices visited by the cycle are probes and at least two of the edges of the cycle are in the same F_i, $1 \le i \le 2$, implying that F_i is not quasitransitive. Thus $(V, F_1 + F_2)$ is acyclic. Similarly $(V, F_1^{-1} + F_2)$ is acyclic. In the following we construct a permutation π such that $G[\pi]$ is an embedding of G. Define two labelings L and L' as follows.

1. Label the vertices in the order determined by a topological sort of vertices of $(V, F_1 + F_2)$, that is, $L(x) = i$ if x is the i^{th} vertex of this sort.
2. Label the vertices according to the order determined by a topological sort of vertices of $(V, F_1^{-1} + F_2)$, that is, that is, $L'(x) = i$ if x is the i^{th} vertex of the new sort.

Then $\pi(i) = L \circ L'^{-1}(i)$, for $i = 1, \ldots, n$.
Notice that $L(y) > L(x)$ if and only if $\overrightarrow{xy} \in F_1 + F_2$. Similarly $L'(y) > L'(x)$ if and only if $\overrightarrow{xy} \in F_1^{-1} + F_2$. Since it is the edges of E which have their

orientations reversed between Steps I and II and $\{x, y\} \not\subseteq \mathbb{N}$ if $(x, y) \in E$, we have

$$(x, y) \in E \iff \{x, y\} \not\subseteq \mathbb{N} \text{ and } (L(x) - L(y))\,(L'(x) - L'(y)) < 0$$

which is exactly what we get by substituting $\pi = L \circ L'^{-1}$ into (6.4). □

By Theorem 6.24 and Theorem 6.28 we obtain:

Theorem 6.29. *A partitioned probe permutation graph can be recognized in* $O(n^3)$ *time.*

7

Partitioned Probe Permutation Graphs

In this chapter, we provide a recognition algorithm for partitioned probe permutation graphs with time complexity $O(n^2)$. We also show that probe permutation graphs have at most $O(n^4)$ minimal separators. As a consequence, for probe permutation graphs there exist polynomial-time algorithms solving problems like TREEWIDTH and MINIMUM FILL-IN. We apply a modular decomposition technique for the recognition of partitioned probe permutation graphs. The recognition of the unpartitioned case remains an open problem. We conjecture that it is polynomial.

Probe permutation graphs are in general not perfect (see Fig. 7.1), but they have many other interesting features. In Section 7.3, we prove that probe permutation graphs have at most $O(n^4)$ minimal separators. An algorithm to find all minimal separators in a graph with polynomial delay appeared in [134]. As a consequence, there exist polynomial-time algorithms solving problems like TREEWIDTH and MINIMUM FILL-IN for probe permutation graphs [24]. Note that the treewidth and pathwidth parameters coincide for permutation graphs [134]. Thus the PATHWIDTH problem, which in general is much more difficult to compute, can also be solved in polynomial time for permutation graphs. It is easy to see that pathwidth and treewidth do not coincide for probe permutation graphs in general.

7.1 Preliminaries

For convenience we define permutation graphs by their matching diagrams.

Definition 7.1. *Let π be a permutation of $(1,\ldots,n)$. The* matching diagram *of π is obtained as follows. Write the integers $(1,\ldots,n)$, horizontally from left to right. Underneath, write the integers $(\pi_1,\ldots,\pi_n)$, also horizontally from left to*

right. Draw n *straight line segments connecting the two* 1*'s, the two* 2*'s, and so on.*

Definition 7.2. *A graph is a* permutation graph *if it is isomorphic to the intersection graph of the line segments of a matching diagram.*

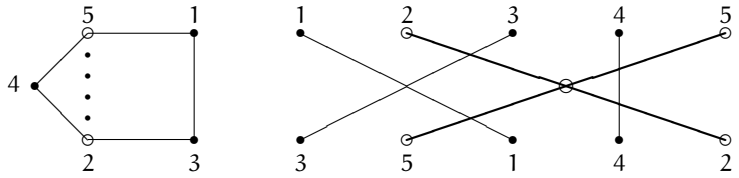


Fig. 7.1. A 5-cycle. Vertices 2 and 5 are nonprobes.

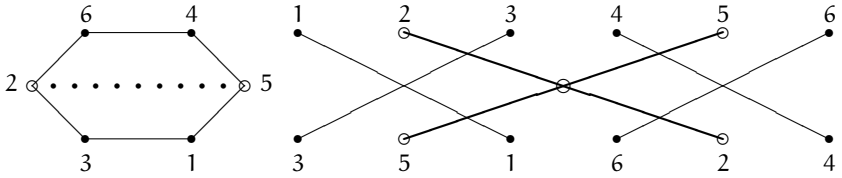


Fig. 7.2. A 6-cycle. Vertices 2 and 5 are nonprobes.

To get acquainted with the class of probe permutation graphs we show that they cannot have induced cycles of length more than 6.

Definition 7.3. *An* asteroidal triple *in a graph* G*, abbreviated* AT*, is a set of three mutually nonadjacent vertices,* $\{x_1, x_2, x_3\}$*, such that removing the neighborhood of any one of the vertices of the triple,* x_i*, leaves the other two vertices of the triple in the same component of* $G - N[x_i]$*. A graph is* AT-free *if it does not contain any* AT*.*

It is easy to see that permutation graphs are AT-free: Consider three parallel lines in a matching diagram. Then any path connecting the outermost two must have some vertex whose line segment crosses the parallel line lying between them.

Theorem 7.4. *A probe* AT*-free graph* G *cannot have chordless cycles of length more than 6. Furthermore, every chordless 6-cycle has exactly two nonprobes at distance 3 in the cycle.*

Proof. Let C be a chordless cycle. Consider the set of probes in C. There can be at most two components in $C - \mathbb{N}$; otherwise there is an AT in any embedding. Hence, if the length of C is more than 5, there must be exactly two components in $C - \mathbb{N}$. Furthermore, each component must be a single vertex or an

edge; otherwise, there still will be an AT in any embedding. Since the set of nonprobes is independent, there must be exactly two nonprobes; otherwise, there are more than two components in $C - \mathbb{N}$. Hence C must have length 6, and the nonprobes must be at distance 3. □

Hence we obtain the following result.

Theorem 7.5. *Probe permutation graphs cannot have induced cycles of length more than 6.*

Theorem 7.6 ([170]). *A graph is a permutation graph if and only if* G *and* $\overline{G}$ *are comparability graphs.*

Notice that permutation graphs form a *self-complementary* class of graphs. That is, if a graph is a permutation graph then so is its complement $\overline{G}$. The class of probe permutation graphs, however, is *not* self-complementary. For example, the disjoint union of two disjoint 6-cycles becomes a permutation graph if one long diagonal is added to each C_6 (Fig. 7.2). In the complement of $2C_6$, the nonprobes necessary to make each $\overline{C}_6$ subgraph a probe permutation graph are not independent in the combined graph. Recall the following definition of the sandwich conjugate:

Definition 7.7. *Let* G *be a partitioned graph. The* sandwich conjugate G^* *of* G *is the partitioned graph with* $\mathbb{P}(G^*) = \mathbb{P}(G)$ *and* $\mathbb{N}(G^*) = \mathbb{N}(G)$ *obtained from* $\overline{G}$ *by removing all edges between vertices of* $\mathbb{N}(G)$.

Note that, if $\mathcal{G}$ is a self-complementary class of graphs, then G is a partitioned probe graph of $\mathcal{G}$ if and only if its sandwich conjugate falls into the same category.
Theorem 7.6 permits permutation graphs to be recognized in linear time, using the linear time algorithm of [146] to find a *modular decomposition tree*.

Definition 7.8. *Let* $G = (V, E)$ *be a graph. A subset* $M \subseteq V$ *of vertices is called a* module *if for every vertex* $z \in V - M$,

$$M \subseteq N(z) \text{ or } M \cap N(z) = \varnothing.$$

i. *A module* M *is called* strong *if for every module* M' *the modules* M *and* M' *do not overlap,* i.e.*, either* $M \cap M' = \varnothing$, $M \subseteq M'$, *or* $M' \subseteq M$.
ii. *A module* M *is* trivial *if* $M = V$, $M = \varnothing$, *or* $|M| = 1$.
iii. *A graph* G *is called* prime *if* G *contains only trivial modules.*

The following celebrated theorem of Gallai started a long line of the research into modular decompositions.

Theorem 7.9 ([145]). *If* G *and* $\overline{G}$ *are connected, then there is a unique partition* P *of* V *and a* transversal set[1] $U \subseteq V$ *such that:*

[1] Sometimes the subgraph induced by the transversal is called the *quotient graph*.

1. $|U| > 3$,
2. $G[U]$ *is a maximal prime subgraph of* G, *and*
3. *for every class* S *of the partition* P, S *is a module and* $|S \cap U| = 1$.

Strong modules satisfy the following property. If $M \neq V$ is a strong module then the smallest strong module M' properly containing M is uniquely determined. This property defines a parent relation in the *modular decomposition tree*.

1. The leaves of this tree are the vertices of the graph.
2. A node is labeled as a *parallel* node if the subgraph induced by the leaves in the subtree is disconnected. The children of the node are the components.
3. A node is labeled as a *series* node if the complement of the subgraph induced by the leaves in the subtree is disconnected. The children are the components of this complement.
4. Finally, an internal node is *prime* if the graph induced by the leaves in the subtree as well as the complement of this induced subgraph are connected. The children are the strong modules mentioned in Theorem 7.9. The node is labeled with the prime graph induced by the transversal set.

In case the graph G is disconnected, we let the transversal set be the subgraph induced by one vertex of each component. In this case the subgraph induced by the transversal set is an independent set. Likewise, when $\overline{G}$ is disconnected, by taking one vertex of each component of $\overline{G}$ we get a transversal set which induces a clique.

The history of algorithms to find the modular decomposition tree is long, starting with [57, 131, 158, 159]. A linear-time algorithm was finally obtained by McConnell and Spinrad, and, independently, by Habib *et al.*, [146]. Much research is still being done to simplify this algorithm. See, *e.g.*, [98] for one of the most recent developments. Using modular decompositions, an orientation of a graph can be obtained in linear time, and it is transitive if the graph is a comparability graph. However, no better algorithm is known for checking the transitivity of the orientation than a matrix multiplication.

For the recognition of permutation graphs the situation is much better. Notice that a diagram for a permutation graph can be obtained as follows. Let F_1 be a transitive orientation of G and let F_2 be a transitive orientation of $\overline{G}$. Then $F_1 + F_2$ is an acyclic orientation of the complete graph; hence it gives a unique linear ordering of the vertices of G. Likewise, $F_1^{-1} + F_2$ gives a unique linear ordering. The first linear ordering can be used for the top line and the second for the bottom line of a permutation diagram representing G. The result is a linear time recognition algorithm for permutation graphs, since

it takes linear time to find F_1 and F_2 using the modular decomposition algorithm of [146], and one only needs to check whether the intersection diagram correctly represents G. The transitivity of F_1 and F_2 is then guaranteed.

Notice that if G is a prime permutation graph, then both G and $\overline{G}$ are uniquely partially orderable, or UPO [86]. That is, F_1 and F_2 are uniquely determined up to the reversal. Therefore, we have the following result.

Lemma 7.10. *A prime permutation graph has a unique matching diagram, up to reversal and exchanging the top and bottom line.*

7.2 Recognition of partitioned probe permutation graphs

Let $G = (\mathbb{P} + \mathbb{N}, E)$ be a partitioned graph. We aim at constructing a diagram for G such that, if x and y are two line segments in the diagram, not both nonprobes, then they cross if and only if the vertices x and y are adjacent in G. Obviously, the graph $G[\mathbb{P}]$ induced by the probes must be a permutation graph, since the class of permutation graphs is hereditary. Using the algorithm of [146], we compute the modular decomposition tree for $G[\mathbb{P}]$. Our strategy is to try to insert the nonprobes into a suitable diagram for $G[\mathbb{P}]$.

Theorem 7.11. *There exists an algorithm to determine in $O(n^2)$ time whether a partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$ is a partitioned probe permutation graph and to obtain an embedding if one exists.*

Proof. We will first describe a recognition algorithm and then analyze the time complexity. We must construct a matching diagram for G such that, if x and y are two lines in the diagram, not both nonprobes, they cross if and only if the vertices x and y are adjacent in G.
Step 1. First we check whether $G[\mathbb{P}]$ is a permutation graph. If $G[\mathbb{P}]$ is not a permutation graph, then G is not a partitioned probe permutation graph.
Step 2. We find the modular decomposition tree T for $G[\mathbb{P}]$. In general, if M is a set of probes, and $x \notin M$ is some other vertex, we say x is *partially adjacent* to M if x is adjacent to one or more vertices of M and also nonadjacent to one or more vertices of M.

Let M be an internal node of T, a strong module of $G[\mathbb{P}]$, and let $\mathbb{N}_M$ be the set of nonprobes which are partially adjacent to M. If a nonprobe is adjacent to no vertex or every vertex of M, then its placement in the diagram, relative to M, is determined at a higher level of T, or if M is the root, the placement is trivial. Let M_i, $1 \leq i \leq r$, be the immediate children of M. Let T_M be the subtree of T with M as the root. Then T_M is the modular tree decomposition of $H = G[M]$. When each M_i is replaced by a single vertex p_i, with adjacency inherited from the module M_i, we obtain the quotient graph H_0, isomorphic

to $H[U]$, the graph induced by the transversal set U (see Theorem 7.9 and following). Since H_0 and $G[M_i]$, $1 \leq i \leq r$, are induced subgraphs of $G[\mathbb{P}]$, they are all permutation graphs.

We start with $M = \mathbb{P}$, the root of T. We repeat Step 3 until we reach every leaf or find that $G = (\mathbb{P} + \mathbb{N}, E)$ is not a partitioned permutation graph. Then Step 4 produces a permutation diagram.

Step 3. Find H_0 and a matching diagram for H_0 so that each nonprobe can be inserted. According to Theorem 7.9 there are three cases to consider, depending on whether H_0 is a clique (the root is a series node), an independent set (the root is a parallel node), or a prime graph with at least four vertices mentioned in the theorem (the root is a prime graph).

We think of the line representing each vertex p_i of H_0 as a thickened line or box, reflecting that each module M_i may have multiple vertices. The endpoints of these lines will thicken to be horizontal line segments, which remain disjoint. We must insert the nonprobe x into the diagram so that it intersects each probe p_i correctly: crossing p_i, disjoint from p_i, or in case of partial adjacency (if $\varnothing \subset (M_i \cap N(x)) \subset M_i$ with proper inclusions), with an endpoint part of the way into p_i. By definition, we need not worry about whether nonprobes cross each other.

Case 1: The node is prime. In this case, both H and $\overline{H}$ are connected. By Theorem 7.9, $H_0 = H[U]$ is a maximal prime permutation subgraph of H. By Lemma 7.10, the matching diagram of H_0 is fixed up to reversing it left to right and flipping it upside down. By definition, if $x \in \mathbb{N}_M$, then M is not a module of $G[M + x]$. Partial adjacency can only occur at the ends of the segment representing x. Thus if x is partially or totally adjacent to every vertex in H_0, the partial adjacency must be at the corners of the diagram, and we easily place x in the diagram of H_0. The only other way that $H_0 + x$, with partial adjacency counted as adjacency, could fail to be prime is if x and p_i have the same adjacency for some vertex p_i in H_0. In that case, x goes "near" p_i. If it has no partial adjacency, except possibly to M_i, then x goes inside, next to, or crossing p_i. There are four distinct vertices in G_0 with diagram endpoints next to those of p_i (or three in case a corner endpoint belongs to p_i). These and M_i are the only possible partial adjacencies of x, and the placement is immediate.

When $H_0 + x$ is prime, the transitive orientation algorithm, as described in [146], correctly orients every edge and nonedge between H_0 and x. We cannot use x as a pivot, but since both H_0 and $H_0 + x$ are prime, the procedure will correctly separate every vertex of $H_0 + x$ without requiring x as a pivot, with one exception. The first run of the algorithm selects some vertex u of H_0 (which is a source or a sink of H_0) to use as the first pivot of the second run. If u was selected because it was the highest ranking probe, but the nonprobe

x was actually higher, then the second run will not distinguish whether x is a source or a sink (having chosen one of the two orientations of H_0). Instead we allow both possibilities. We check which one places x correctly in the diagram of H_0. Once we have placed x in the diagram, the possible partial adjacencies are on either side of the initial placement of the endpoints of x.

Case 2: The node is a parallel node; that is, $H = G[M]$ is disconnected. Each M_i is a component of H, and H_0 is an independent set. The vertices of H_0 correspond to the components of H. In this case the p_i's must be in a sequence, so that each nonprobe is adjacent to consecutive probes in the diagram of H_0, with partial adjacency only allowed on the ends. To find this ordering, let $S = \{p_1, \ldots, p_r\}$. We first construct a collection of subsets $\mathcal{C}$ of S as follows. For each $x \in \mathbb{N}_M$, we include in $\mathcal{C}$ the subset $S_x = \{p_i | x \text{ is adjacent to a vertex of } M_i\}$. The nonprobe x may have partial adjacency to up to two probes in H_0. In that case we also include in $\mathcal{C}$ the subset $S_x - p_i$ for each p_i if x is partially adjacent to M_i.

We also need to ensure that if a nonprobe has adjacency outside M, then its neighbors inside M are placed together on the correct side of the diagram. To this effect, we include two artificial probe vertices in S, $p^\flat$ and $p^\sharp$, and in constructing $\mathcal{C}$ make $p^\flat$ adjacent to every nonprobe which enters M from the left, while making $p^\sharp$ adjacent to every nonprobe which enters M from the right. To force $p^\flat$ and $p^\sharp$ to the outside, we include in $\mathcal{C}$ the subsets $M + p^\flat$ and $M + p^\sharp$. Note that left and right here are relative designations derived from the parent of M.

It is clear that we need to find a permutation π of the elements of S such that for every subset $I \in \mathcal{C}$, the elements of I appear as a consecutive subsequence of π. Booth and Luecker [23] showed that this problem can be solved (using the PQ-tree) in $O(|\mathcal{C}| + |S| + \sum_{I \in \mathcal{C}} |I|)$ time. The permutation π gives a matching diagram of H_0.

Case 3: The node is a series node; that is, H_0 is a clique. In this case we consider the sandwich conjugate and the diagram of the parallel node $\overline{H}_0$, which is obtained by reversing the bottom line. Those nonprobes which have the top end inside the diagram of H_0 and the bottom outside will reverse their direction of exit, while those with the bottom end inside H_0 and the top end outside do not.

We should note that in Case 2 we could assume that the parent node of M was prime, because the parent of a parallel node cannot be another parallel node, and if the parent were a clique we would have taken the conjugate. In a prime node we have the exact information as to which nonprobes exit a child node to the left, exit to the right, or may be considered internal to the child node.

When M is a series node and the parent is a parallel node, we have to use a second method to determine whether a nonprobe enters the diagram of $\overline{H}_0$ from the left or the right. Determine a shortest adjacency list in $\overline{H}_0$ of any nonprobe whose line exits the diagram. The left-right partition is made according to whether the adjacency list of a nonprobe contains this short list as a subset or does not contain it.

In this case we get a relative up-down orientation of a nonprobe depending on whether or not the right-left orientation changed when we took the sandwich conjugate. The rest of the algorithm is similar to Case 2.

Step 4. We create the matching diagram for H_0. First we have to get the diagrams of all the nodes right side up. Designate the internal nodes of T by M_i, $1 \leq i \leq s$. We have to determine how to flip the diagram of each M_i over so that every nonprobe gets an endpoint on the top line and the other endpoint on the bottom line. We do so by reducing the problem to finding a proper 2-coloring of a bipartite graph R. Each M_i is represented by two vertices t_i and b_i. These are made adjacent in R to make sure they get assigned to different lines of the diagram, *i.e.,* one to the top line and the other to the bottom line. Consider a nonprobe x which partially intersects two modules M_i and M_j, so that one endpoint is in each. Suppose we know from Case 1 or from Case 3 where each endpoint goes with respect to t_i, b_i, t_j, and b_j. Then we connect the two vertices in R, one of $\{t_i, b_i\}$ and one of $\{t_j, b_j\}$, where the nonprobe has its endpoints. (In case we have information about only one endpoint of a nonprobe, we can connect the known locations of that endpoint to the opposite vertex, t_i or b_i, of one of the known locations.) The flip-over problem reduces to finding a proper 2-coloring of R. If the problem has no solution, then G is not a partitioned probe permutation graph.

Finally we assemble the diagram starting from the root of T, recursively replacing each box in the diagram by the diagram of the node it represents, repeatedly refining the placement of the nonprobes.

The correctness of the algorithm is clear. Either we obtain a matching diagram for G such that if x and y are two lines in the diagram, at least one is a probe, they cross if and only if x and y are adjacent in G, or else we discover that G is not a partitioned probe permutation graph.

Next, we analyze the complexity of the algorithm. Steps 1 and 2 take $O(n+m)$. We obtain the modular decomposition tree T of $G[\mathbb{P}]$ and the unique diagram of each prime node. The transitive orientation algorithm [146] simultaneously gives the correct placement of the nonprobes in prime nodes, Case 1 of Step 3, in the same time. We can clearly determine the nodes to which each nonprobe x is partially adjacent in $O(n)$ time.

For Cases 2 and 3 of Step 3, we have the PQ-algorithm which takes $O(|\mathcal{C}|+|S|+\sum_{I\in\mathcal{C}}|I|)$ for each node. Since $(|\mathcal{C}|+|S|)$ is within $O(n)$, and the number of

nodes is within $O(n)$, the contribution of the first two terms is within $O(n^2)$. For each nonprobe x, then, we need to show that the sum of its adjacency lists $|I|$, in nodes where x has partial adjacency, is no more than $O(n)$. Note that $|I|$ is not the number of vertices in a node M to which x is adjacent, but the number of children of M to which x is at least partially adjacent. If a node is totally adjacent to x, then the descendants of that node do not contribute. Since a node is a nonempty module, the contribution of total adjacencies to the sum of $|I|$ for x cannot exceed n. Likewise, x can have at most two partial adjacencies, and only one node can have that many. Therefore the contribution of partial adjacencies of x to the sum of $|I|$ is also less than n.

In Case 3 of Step 3, we determine which end of a nonprobe goes up in constant time for each nonprobe at each node. In Step 4 we 2-color a bipartite graph R in which there are at most $2n$ vertices. We said we would connect two vertices in R if they represent the locations of opposite ends of the line representing some nonprobe in two nodes of the tree T. The information obtained from a single nonprobe would be represented as a complete bipartite graph on certain vertices of R. It could require checking $O(n^2)$ edges of R for each nonprobe, inserting the edge if it is not already represented. In fact, the information obtained from each nonprobe can be represented by a spanning tree of this complete bipartite graph. Therefore, we can construct R by checking $O(n)$ edges for each nonprobe. Finally, assembling the diagram can be done in linear time, and checking it for correctness can be done in $O(n^2)$ time. Therefore recognition of partitioned probe permutation graphs, and construction of a diagram for an embedding if a graph is a partitioned probe permutation graph, can be accomplished in $O(n^2)$ time. □

Conjecture 7.12. There exists a polynomial-time algorithm for the recognition of (unpartitioned) probe permutation graphs.

7.3 Treewidth of probe permutation graphs

In this section we analyze the structure of the minimal separators in an unpartitioned probe permutation graph, proving that problems such as TREEWIDTH and MINIMUM FILL-IN are computable in polynomial time for graphs in this class. Although at the moment we do not know how to recognize this class, we show that the treewidth can be determined nevertheless, or else a conclusion is drawn that the graph is not in the class, if more than $n^4/4$ minimal separators are found. Assume throughout that G is connected.

Definition 7.13. *Let* $G = (V, E)$ *be a graph.*

1. *A set* $\Omega \subset V$ *is a* separator *if* $G - \Omega$ *has at least two components.*

2. *If* Ω *is a separator and* C *is a component of* $G - \Omega$ *such that every vertex of* Ω *has at least one neighbor in* C*, then* C *is a* full component *of* Ω.
3. *A separator* Ω *is a* minimal separator *if it has at least two full components.*
4. *If* x *and* y *are vertices in different full components of* Ω*, then* Ω *is called a* minimal x, y-separator.

Definition 7.14. *Consider a matching diagram. A* scanline *in the diagram is any line segment with one end vertex on each horizontal line such that the endpoints do not coincide with endpoints of line segments of the diagram.*

Consider a scanline s in a matching diagram such that at least one line segment x of the diagram has its two endpoints to the left and at least one line segment y has both its endpoints to the right of s. Take out all the lines in the matching diagram that *cross* the scanline s. The corresponding set of vertices clearly separates x and y into different components. For minimal separators, the converse also holds:

Theorem 7.15 ([20]). *Let* G *be a permutation graph and consider any matching diagram of* G*. Let* x *and* y *be nonadjacent vertices in* G*. Then every* minimal x, y*-separator consists of all line segments that cross some scanline that lies between the line segments of* x *and* y *in the diagram.*

Corollary 7.16. *A permutation graph has fewer than* n^2 *minimal separators.*

Theorem 7.17. *Let* G *be a connected probe permutation graph equipped with an embedding* H*. Consider a matching diagram of* H*. Let* Ω *be a minimal separator in* G*. Then there exist* noncrossing *scanlines* s_1 *and* s_2*, such that*

1. $\Omega_{\mathbb{P}}$ *consists of all the probes crossing at least one of* s_1 *and* s_2*, or with both endpoints between* s_1 *and* s_2*, and*
2. $\Omega_{\mathbb{N}}$ *consists of all the nonprobes that cross* both s_1 *and* s_2.

Proof. Let Ω be a minimal x, y-separator in G. If $x \in \mathbb{N}$ and $\Omega = N(x)$, then Ω consists of only probes. We can take one scanline immediately next to x and the other scanline just outside the last vertices of the diagram. The argument is similar when $\Omega = N(y)$ and $y \in \mathbb{N}$.

The other possibility is that the two full components of $G - \Omega$ containing x and y each contain at least one probe. Assume that the probes of the component C_x that contains x appear to the left of the probes of the component C_y that contains y.[2] Take a scanline s_1 immediately to the right of the probes of C_x. Also consider the probes of $G - \Omega$, including the probes of C_y, which appear to the right of the probes of C_x. Take scanline s_2 immediately to the left of these probes. Then Ω contains all the probes between s_1 and s_2 or

[2] Clearly they cannot “interlace."

crossing at least one of them. All the nonprobes of Ω cross both of s_1 and s_2. Since this set separates x and y and since Ω is minimal, Ω is equal to it. □

Since the scanlines do not cross, and except for the $|\mathbb{N}|$ neighborhood separators of nonprobes, the endpoints are in the interior of the diagram, we have this corollary.

Corollary 7.18. *A probe permutation graph has fewer than* $n^4/4$ *minimal separators.*

Theorem 7.19. *Polynomial-time algorithms exist solving the* TREEWIDTH *and the* MINIMUM FILL-IN *problems for the class of probe permutation graphs.*

Proof. An algorithm that finds all minimal separators in a graph with polynomial delay appeared in [134]. Bouchitte and Todinca showed in [24] how to solve the problems mentioned above in polynomial time, given the list of minimal separators.

Conjecture 7.20. There is a polynomial-time algorithm to solve the PATHWIDTH problem for the class of probe permutation graphs.

8

Partitioned Probe Distance Hereditary Graphs

In this chapter we give two recognition algorithms for partitioned probe distance-hereditary graphs. Both algorithms run roughly in $O(n^4)$ time. The first one is theoretically slightly faster. The second one has the advantage of producing a decomposition tree.

8.1 Preliminaries

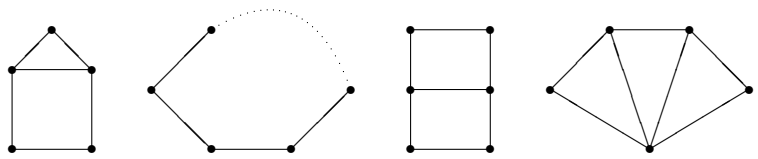


Fig. 8.1. A house, a hole, a domino, and a gem.

Definition 8.1 ([118]). *A connected graph is* distance hereditary *if the distance between any two vertices remains the same in every connected induced subgraph.*[1]

Definition 8.2. *A* pendant vertex *in a graph is a vertex of degree 1.*

Definition 8.3. *A* twin *in a graph is a module with two vertices. A twin is* true *if the vertices are adjacent. Otherwise the twin is* false.

Theorem 8.4 ([6]). *Connected distance-hereditary graphs with at least two vertices are exactly the graphs that can be evoked from an edge by one-vertex extensions consisting of attaching pendant vertices and creating twins.*

[1] Interpretive, we call a graph also distance hereditary if every component is such.

Remark 8.5. This implies that distance-hereditary graphs are contained in the class of circle graphs.[2]

To start with, distance-hereditary graphs can be captured by forbidden induced subgraphs [6, 59]. For the house, hole, domino, and the gem, we refer to Figure 8.1. Note that, if a graph does not contain an induced house, hole, domino, nor gem, then adding a pendant vertex, or creating a twin cannot give rise to one of these induced subgraphs.

Theorem 8.6 (Bandelt & Mulder [6]). [3] *Let* G *be a graph. The following conditions are equivalent:*

1. G *is connected and distance hereditary.*
2. G *is connected and does not contain a house, hole, domino, or a gem as an induced subgraph.*
3. *Every connected induced subgraph of* G *with at least 2 vertices has a pendant vertex or a twin.*
4. *For every pair of vertices* x *and* y *with* $d(x,y) = 2$*, there is no induced* x,y*–path of length greater than 2.*

Affirmation of membership in the class of distance-hereditary graphs can be obtained using a linear time algorithm [64, 162]. They have a nice characterization in terms of a certain decomposition tree [38, 64, 122] which allows efficient algorithms for many problems, see, e.g., [120, 51, 163]. This decomposition tree can be obtained in linear time [36, 64].

Notice that distance-hereditary graphs cannot have antiholes, since this would imply the existence of an induced hole or an induced $\overline{P}_5$, which is a house. This demonstrates that they are perfect.

Theorem 8.7 ([118]). *A graph is distance hereditary if and only if every cycle of length at least 5 has at least 2 crossing chords.*

Recall that Meyniel graphs are those graphs in which every cycle of length at least 5 has at least 2 chords. *A fortiori* distance-hereditary graphs are Meyniel. By Theorem 2.6 on page 14 we obtain that probe distance-hereditary graphs are slim, hence perfect.

We present another proof showing that probe distance hereditary graphs are perfect. Recall that the class of weakly chordal graphs is defined as the set of graphs without an induced hole or antihole.

---

[2] Chronicled in [26], this was observed by Damaschke. A *circle graph* is the intersection graph of a set of chords in a plane circle.

[3] See also [101]. The linear time recognition algorithm described in [101] based on item 2 in Theorem 8.6 seems to contain an error, which was corrected in [64].

Theorem 8.8. *Probe distance-hereditary graphs are weakly chordal. The two classes do not coincide.*

Proof. Let C be a hole. If C has two adjacent probes, in any embedding the two neighbors must be nonprobes that are adjacent in any embedding, otherwise there is a hole in the embedding. If C is a C_5, this creates a house. Otherwise, the two other neighbors of the nonprobes must be probes. Thus either a domino, a house or a hole is created.
Assume the probes and nonprobes alternate. Hence C is an even hole. Consider two probes at distance 2. Hence, in any embedding, all induced paths must be of length 2. But this is impossible, since the other path between the probes in C contains at least three edges in any embedding.
Since an antihole has independence number 2, it can have at most two nonprobes. Adding an edge to the antihole leaves (at least) a house since deleting an edge in a hole leaves a path of length at least 5 which is the complement of a house.

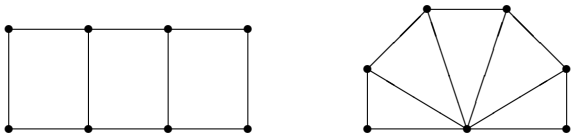


Fig. 8.2. Two weakly chordal graphs which are not probe distance hereditary.

We show that the weakly chordal graphs depicted in Figure 8.2 are not probe distance hereditary.
Consider the graph on the left in Figure 8.2. By Lemma 8.12 below, a domino must have exactly two nonprobes which are at distance 3. There is no possible choice for the nonprobes in the graph on the left in Figure 8.2.
The universal vertex in the graph on the right must be a probe by Corollary 8.10 below and its neighborhood must be embedded as a cograph. By Theorem 3.2 on page 22 a P_6 is not a probe cograph. □

To get acquainted with the class of probe distance-hereditary graphs we start with some easy observations.

Lemma 8.9. *If* G *is a probe distance-hereditary graph, and if* G *has a vertex* x *such that its neighborhood* $N(x)$ *is* not *a cograph, then* x *must be a probe in any distance-hereditary embedding of* G.

Proof. Assume G is probe distance hereditary. If x is a nonprobe, then all its neighbors must be probes. Hence $N(x)$ must be P_4-free, otherwise $N[x]$ contains an induced gem in any embedding. □

Corollary 8.10. *The universal vertex of a gem is a probe.*

The next lemma shows that also any induced house can be assigned a designated vertex.

Lemma 8.11. *If* G *has an induced house, then the simplicial vertex in the house is a nonprobe.*

Proof. A house can have only two nonprobes. If they are two vertices of the square, then adding an edge creates a gem, which is a contradiction. □

Lemma 8.12. *If a probe distance-hereditary graph has an induced domino* D *then in any embedding of* G*,* D *has exactly two vertices at distance 3 which are nonprobes.*

Proof. Any maximal independent set in a domino has either 2 or 3 vertices. Assume it has 3 nonprobes. Then the three nonprobes have pairwise distance 2. If only one edge is added, this creates a house. If two or three edges are added in the embedding it creates a house or a gem. Hence a domino has two nonprobes. If they are at distance 2, a house is created in the embedding. Hence there are exactly two nonprobes at distance 3. □

Corollary 8.13. *If* G *is probe distance hereditary and* D *is an induced domino then the two vertices that have degree 3 in* D *are probes.*

8.2 A partitioned probe Bandelt & Mulder

Recall from Theorem 8.4 that a distance-hereditary graph with at least an edge has either a pendant vertex or a twin.

Definition 8.14. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph. A pair of vertices* $\{x, y\}$ *is a* probe twin *if either:*

1. $x, y \in \mathbb{P}$ *and* $\{x, y\}$ *is a module in* G*, or*
2. $x, y \in \mathbb{N}$ *and* $\{x, y\}$ *is a module in* G*, or*
3. $x \in \mathbb{P}$*,* $y \in \mathbb{N}$ *and* $N(y) - x = (N(x) - y) \cap \mathbb{P}$*.*

Lemma 8.15. *Assume* x *is a pendant vertex in* G*. Then* G *is probe distance hereditary if and only if* $G - x$ *is probe distance hereditary.*

Proof. Consider an embedding of $G - x$ into a distance-hereditary graph. Obviously, adding x as a pendant vertex to the embedding does not introduce a house, hole, domino, or gem. □

Theorem 8.16 (The Partitioned Probe Bandelt & Mulder). *A partitioned graph* $G = (\mathbb{P}+\mathbb{N}, E)$ *is partitioned probe distance hereditary if and only if every connected induced subgraph with at least 2 vertices has a pendant vertex or a probe twin.*

Proof. Assume that G is partitioned probe distance hereditary. We may assume that G is connected. We show that G has a pendant vertex or a probe twin. Let H be an embedding of G. Assume H has a pendant vertex x. Since G is connected, x is also pendant in G. If H has a twin, say $\{x, y\}$, then $\{x, y\}$ is obviously a probe twin in G.
Assume that every connected induced subgraph of G has a pendant vertex or a probe twin. Let x be a pendant vertex in G. By induction $G-x$ is partitioned probe distance hereditary. By Lemma 8.15 G is also partitioned probe distance hereditary.
Assume G has a twin $\{x, y\}$. If at least one of them is a nonprobe then let y be a nonprobe. By induction $G - y$ is partitioned probe distance hereditary. We may add y to an embedding of $G-y$ as a twin of x. Since this does not create a house, hole, domino, or gem, the graph is an embedding of G.
Assume G has a probe twin $\{x, y\}$ which is not a twin. Assume $x \in \mathbb{P}$ and $y \in \mathbb{N}$. By induction $G - y$ is partitioned probe distance hereditary. Let H' be an embedding of $G - y$. Replacing y by the twin $\{x, y\}$ in H creates an embedding of G. □

Theorem 8.16 delivers the goods for a fast recognition algorithm.

Theorem 8.17. *There exists an* $O(n(n+m))$ *time algorithm for the recognition of partitioned probe distance-hereditary graphs.*

Proof. Obviously, finding a pendant vertex takes only linear time. Consider the problem of finding a true twin in G. Sort the *closed* neighborhoods of the vertices in lexicographic order. This can be done with a RADIX SORT in $O(n + m)$ time (see, *e.g.*, [157, Theorem 1 on page 84] or [48, Chapter 8]. The closed neighborhoods of a true twin will appear consecutively in this order. Finding a false twin can be done by sorting the open neighborhoods of the vertices. A probe twin with vertices $x \in \mathbb{P}$ and $y \in \mathbb{N}$ can be found by sorting the neighborhoods consisting of probes. □

In all probability there is a more efficient algorithm, using results of, *e.g.*, [64]. Instead of tracking this down we present an alternative recognition algorithm in Section 8.4.

Remark 8.18. Obviously, by Theorem 8.6, cographs are distance hereditary. Cographs can be characterized as those graphs for which every nontrivial induced subgraph has a twin [26, Theorem 11.3.3]. It easily follows that there

exists an $O(n(n+m))$ algorithm to recognize partitioned probe cographs. This slightly improves the result of Theorem 3.7 on page 24.

8.3 Partitioned probe ptolemaic graphs

Definition 8.19 ([130]). *A connected graph is* ptolemaic *if for every 4 vertices* x, y, u, *and* v*:*

$$d(x,y)d(u,v) \leq d(x,u)d(y,v) + d(x,v)d(y,u)$$

A graph is ptolemaic *if every component is ptolemaic.*

Ptolemaic graphs can be characterized as those chordal graphs in which every 5-cycle has at least 3 chords, or, as those chordal graphs in which all chordless paths are shortest paths [26, 72, 180]. We will use the following characterization of ptolemaic graphs.

Theorem 8.20 ([119]). *A graph is ptolemaic if and only if it is distance hereditary and chordal.*

Since holes, houses, and dominos are not chordal, ptolemaic graphs are those chordal graphs which are gem-free. Since every sun has a gem, they are properly contained in the class of strongly chordal graphs.

Lemma 8.21. *Assume* G *is ptolemaic and let* x *be a vertex in* G*. The graph* F *obtained from* G *by adding edges such that* $N(x)$ *becomes a clique is ptolemaic.*

Proof. Obviously, making a clique of $N(x)$ does not create a chordless cycle since G is chordal. Assume that F has a gem S. If x is a vertex of S then it must be a simplicial. Deleting the edge between the two neighbors in S creates a chordless cycle. If x is not in S then it is adjacent to a triangle or an edge in S. Then deleting the edge creates a chordless cycle or gem with vertex x. □

Definition 8.22. *Let* $G = (\mathbb{P} + \mathbb{N}, E)$ *be a partitioned graph. A pair of vertices* $\{x,y\}$ *is an* ptolemaic twin *if either:*

1. $x, y \in \mathbb{P}$ *and*
   i. $\{x,y\}$ *is a module in* G *and*
   ii. *if* x *and* y *are not adjacent then* $N(x)$ *is a probe clique, or*
2. $x, y \in \mathbb{N}$ *and* $\{x,y\}$ *is a module, or*
3. $x \in \mathbb{P}$ *and* $y \in \mathbb{N}$ *and*
   i. $N(y) - x = (N(x) - y) \cap \mathbb{P}$ *and*
   ii. *if* x *and* y *are not adjacent then* $N(x)$ *is a probe clique.*

We now easily obtain the following characterization.

Theorem 8.23. *A partitioned graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *is partitioned probe ptolemaic if and only if every connected induced subgraph of* G *with at least 2 vertices has a pendant vertex or a ptolemaic twin.*

Proof. Assume G is partitioned probe ptolemaic and let H be an embedding of G. Let C be the vertex set of a connected induced subgraph of G and assume $|C| \geq 2$. By Theorem 8.20, the class of ptolemaic graphs is hereditary thus also $H[C]$ is ptolemaic. If x is a pendant vertex in $H[C]$ then, since $G[C]$ is connected, x is also a pendant vertex of $G[C]$. Assume $H[C]$ has a twin, say $\{x, y\}$. Notice that, if x and y are not adjacent then $N_H(x) = N_H(y)$ is a clique since the graph is chordal. It is easy to check that $\{x, y\}$ is a ptolemaic twin in $G[C]$.
Assume that every connected induced subgraph of G has a pendant vertex or a ptolemaic twin. We may assume that G is connected. Let x be a pendant vertex in G. By induction $G - x$ has an embedding as a ptolemaic graph. Since a gem or chordless cycle has no pendant vertex we may add x to the embedding.
Assume G has a ptolemaic twin $\{x, y\}$ with $x, y \in \mathbb{P}$. By induction, the graph $G - y$ can be embedded as a ptolemaic graph. If x and y are adjacent then we add y as a twin of x to this embedding. This creates no chordless cycle nor any gem. If x and y are nonadjacent, then by Lemma 8.21 we may make x simplicial in this embedding. We add y as a twin of x.
Assume G has a ptolemaic twin $\{x, y\}$ with $x, y \in \mathbb{N}$. By induction, the graph $G - y$ has an embedding into a ptolemaic graph. We add y to the embedding as a true twin of x. This cannot create a gem or a chordless cycle.
Assume G has an ptolemaic twin with $x \in \mathbb{P}$ and $y \in \mathbb{N}$. Consider an embedding of $G - y$. If x and y are connected we can add y as a true twin of x. Assume x and y are not connected. Let F be an embedding of $G - y$. Then by Lemma 8.21 we may make a clique of $N_F(x)$. Add y as a false twin of x. □

Theorem 8.24. *There exists an* $O(n^3)$ *algorithm for the recognition of partitioned probe ptolemaic graphs and for finding an embedding if there exists one.*

Proof. Using a RADIX SORT a pendant vertex or a ptolemaic twin in a partitioned graph can be found in linear time. □

8.4 Recognition of PPDH–graphs

Without much ado we carry on with our next recognition algorithm. If X is a subset of vertices in a partitioned graph $G = (\mathbb{P} + \mathbb{N}, E)$ then we write $\mathbb{P}(X) = X \cap \mathbb{P}$ and $\mathbb{N}(X) = X \cap \mathbb{N}$.

Definition 8.25 ([144]). *Let* $G = (V, E)$ *be a connected graph, and let* $V = V_1 + V_2$ *be a partition of* V *such that* $|V_1|, |V_2| > 1$. *We call* (V_1, V_2) *a* split *of* G *if* $N(V_1)$ *and* $N(V_2)$ *are completely adjacent.*

If G has no split, G is called *prime* with respect to split decomposition. The *split-decomposition tree* of G is formed by choosing any split (V_1, V_2) of G and two vertices $v_1 \in N(V_1)$ and $v_2 \in N(V_2)$, and decomposing recursively $G[V_1 + v_1]$ and $G[V_2 + v_2]$. Ma and Spinrad noted that distance-hereditary graphs are completely decomposable with respect to split decomposition [144]:

Lemma 8.26 ([144]). *Let* G *be a connected distance-hereditary graph with at least 4 vertices. Then* G *has a split.*

In the following we define the *probe split* of a partitioned probe graph as a generalization of the split of a graph. We will use the probe split of partitioned probe graphs to recognize partitioned probe distance-hereditary graphs. As before, we say that two disjoint subsets of vertices A and B are completely adjacent if every vertex of A is adjacent to every vertex of B.

Definition 8.27. *Let* G *be a connected, partitioned graph with* $|V| > 3$. *Let* $V = V_1 + V_2$ *be a partition of* V *such that* $|V_1|, |V_2| > 1$. *We call* (V_1, V_2) *a* probe split *of* G *if*

1. $N(V_1)$ *and* $\mathbb{P}(N(V_2))$ *are completely adjacent, and*
2. $N(V_2)$ *and* $\mathbb{P}(N(V_1))$ *are completely adjacent.*

The following proposition is immediate from the above definition.

Proposition 8.28. *Let* G *be a connected partitioned probe graph. If* (V_1, V_2) *is a probe split of* G *such that* $v_1 \in \mathbb{P}(V_1)$, $v_2 \in \mathbb{P}(V_2)$, *and* $(v_1, v_2) \in E$, *then for all* $p \in \mathbb{P}(V_1)$ *and* $q \in V_2$, $(p, q) \in E$ *if and only if* $(v_1, q), (p, v_2) \in E$.

Let G be a connected partitioned probe distance-hereditary graph with $|V| > 3$. Let H be an embedding of G. Because G is connected and H is obtained from G by adding edges between nonprobes in G, also H is connected. By Lemma 8.26, H has a split (V_1, V_2). We also have that every vertex in $\mathbb{P}(N_G(V_1))$ is adjacent to every vertex in $N_G(V_2)$ and every vertex in $\mathbb{P}(N_G(V_2))$ is adjacent to every vertex in $N_G(V_1)$. Thus (V_1, V_2) is a probe split of G. We have the following lemma.

Lemma 8.29. *Let* $G = (V = \mathbb{P} + \mathbb{N}, E)$ *be a connected partitioned probe distance-hereditary graph with* $|V| > 3$. *Then* G *has a probe split.*

Lemma 8.30. *Let* G_1 *and* G_2 *be two distance-hereditary graphs, with* $v_1 \in G_1$ *and* $v_2 \in G_2$. *Let* G *be the graph with* $V(G) = V(G_1) + V(G_2) - \{v_1, v_2\}$ *and* $E(G) = E(G_1 - v_1) + E(G_2 - v_2) + \{(x, y) \mid x \in N_{G_1}(v_1) \text{ and } y \in N_{G_2}(v_2)\}$. *Then* G *is a distance-hereditary graph.*

Proof. Let H be any connected induced subgraph of G containing vertices u and v. For simplicity, let V_1 denote $V(G_1) - v_1$ and let V_2 denote $V(G_2) - v_2$. We must show that the distances from u to v are the same in G and in H.
Suppose $u, v \in V_1$. Because $N(V_1)$ and $N(V_2)$ are completely adjacent to each other, the shortest path connecting u and v, in G or in H, visits V_2 at most once, say at $x \in V_2$ or at $x' \in V_2$, respectively. Since G_1 is distance hereditary, and G_1 is isomorphic to $G[V_1 + x]$ and to $G[V_1 + x']$, the path lengths must be equal.
Similarly, a shortest path from $u \in V_1$ to $v \in V_2$ in G must visit each of $N(V_2)$ and $N(V_1)$ exactly once, say at $y \in N(V_2)$ and at $x \in N(V_1)$. Suppose the shortest path in H visits $y' \in N(V_2)$ and $x' \in N(V_1)$. Again since $G_1 \cong G[V_1 + x] \cong G[V_1 + x']$, the paths from u to x in G and from u to x' in H have equal length. The same argument applies to the paths from y to v in G and from y' to v in H. Therefore the path from u to v in H has the same length as the shortest path in G. □

Lemma 8.31. *Let G be connected and assume that G is partitioned probe distance hereditary. Let (V_1, V_2) be a probe split of G. For $i = 1, 2$, let $v_i \in \mathbb{P}(N(V_i))$ if $\mathbb{P}(N(V_i)) \neq \varnothing$ or let $v_i \in \mathbb{N}(N(V_i))$ otherwise. Then:*

1. *$G[V_1 + v_1]$ and $G[V_2 + v_2]$ are partitioned probe distance hereditary with their probe and nonprobe sets inherited from G.*
2. *Let H_1 and H_2 be embeddings of $G[V_1 + v_1]$ and $G[V_2 + v_2]$, respectively, and let H be the graph with $V(H) = V(G)$ and $E(H) = E(H_1) \cup E(H_2) \cup E^*$ where $E^* = \{(x, y) \mid x \in N_{H_1}(v_1)$ and $y \in N_{H_2}(v_2)\}$. Then, H is an embedding of G.*

Proof. The first statement follows from the fact that being a partitioned probe graph of some hereditary graph class $\mathcal{G}$ is a hereditary property. We prove the second statement. By Lemma 8.30, H is distance hereditary. For simplicity, let G_i denote $G[V_i + v_i]$. By definition, $N_{G_i}(v_i) = N_G(V_{3-i})$ for $i = 1, 2$. In H, by definition of E^*, every vertex in $N_G(V_1)$ is adjacent to every vertex in $N_G(V_2)$. Since (V_1, V_2) is a probe split of G, every edge in $E^* - E(G)$ connects a vertex in $\mathbb{N}(N_G(V_1))$ and a vertex in $\mathbb{N}(N_G(V_2))$. Hence we only add edges between two vertices in $\mathbb{N}(G)$ to obtain H from G. □

To test whether a connected partitioned graph G is probe distance hereditary we first try to find a probe split of G. If G has no probe split, then G is not partitioned probe distance hereditary by Lemma 8.29. Suppose G has a probe split (V_1, V_2). As stated in Lemma 8.31, for $i = 1, 2$, we choose vertices $v_i \in \mathbb{P}(N_G(V_i))$ if $\mathbb{P}(N_G(V_i)) \neq \varnothing$ or $v_i \in \mathbb{N}(N_G(V_i))$ otherwise, and test recursively whether $G[V_1 + v_1]$ and $G[V_2 + v_2]$ are each partitioned probe distance hereditary, with the inherited probe and nonprobe sets. If either of

them is not partitioned probe distance hereditary, then G is not partitioned probe distance hereditary by Proposition 8.28. If both of them are partitioned probe distance hereditary, then by Lemma 8.31, we can obtain an embedding of G from embeddings of $G[V_1 + v_1]$ and $G[V_2 + v_2]$. Hence G is a partitioned probe distance-hereditary graph.
We obtain the following theorem.

Theorem 8.32. *Let* G *be a partitioned probe graph.*
1. *If* G *is disconnected, then it is a partitioned probe distance-hereditary graph if and only if the subgraph induced by every connected component is partitioned probe distance hereditary.*
2. *If* G *is connected with at least four vertices, then it is a partitioned probe distance-hereditary graph if and only if* G *has a probe split* (V_1, V_2) *and both* $G[V_1 + v_1]$ *and* $G[V_2 + v_2]$ *as defined in Lemma 8.31 are partitioned probe distance hereditary.*

Suppose we can determine whether a partitioned, connected graph G has a probe split, and if so find one, in $O(T(n))$ time. Then the recognition of partitioned probe distance-hereditary graphs can be done in $O(n \times T(n))$ time, since a connected partitioned probe distance-hereditary graph can be decomposed into 3-vertex prime graphs in $n - 3$ decompositions. An embedding of G can be computed in $O(n \times T(n) + n^2)$ if G is partitioned probe distance hereditary.

We show that the algorithm given by Cunningham [58] to find a split of a directed graph can be modified to find a probe split of a partitioned probe graph in $O(n^3)$ time if it exists.

Suppose (V_1, V_2) is a probe split of G. Since G is connected, at least one of $\mathbb{P}(N_G(V_1))$ and $\mathbb{P}(N_G(V_2))$ is not empty. Suppose one of them is empty. Then (V_1, V_2) is a split of G. On the other hand, suppose neither $\mathbb{P}(N_G(V_1))$ nor $\mathbb{P}(N_G(V_2))$ is empty. Let F be a spanning forest of $G[\mathbb{P}(G)]$ with the minimum number of components. Since $\mathbb{P}(N_G(V_1))$ and $\mathbb{P}(N_G(V_2))$ belong to the same component of $G[\mathbb{P}(G)]$, there exists an edge $(v_1, v_2) \in E(F)$ with $v_1 \in \mathbb{P}(N_G(V_2))$ and $v_2 \in \mathbb{P}(N_G(V_1))$. For a vertex $z \in V(G) - \{v_1, v_2\}$, either $z \in V_1$ or $z \in V_2$. Thus we can reduce the problem of finding a probe split to a problem referred to as Problem PS: given an edge $(v_1, v_2) \in E(F)$ (recall $v_1, v_2 \in \mathbb{P}$), pick another vertex $z \in V - \{v_1, v_2\}$ and find, if one exists, a probe split (V_1, V_2) with V_1 containing v_1 and z, and with V_2 containing v_2. Algorithm 3 sketches the reduction and Algorithm 4 gives the details of an algorithm for solving Problem PS.

To present Algorithm 4, we will make use of some terminology used in [58]. If $(v_1, v_2) \in E$ and $p, q \in V$, we say that $P(v_1, v_2, p, q)$ holds if the following condition holds.

Algorithm 3 Finding a probe split for a partitioned probe graph.

Input: A partitioned connected probe graph G with $|V(G)| \geq 4$.
Output: A probe split (V_1, V_2) of G if one exists.

```
1: if G has a split then
2:    Find a split (V1, V2) of G;
3:    output (V1, V2) as a probe split of G and exit;
4: else
5:    Find a spanning forest F of G[ℙ(G)];
6:    for each edge (v1, v2) ∈ E(F) do
7:       arbitrarily choose a vertex z ∈ V(G) − {v1, v2}
8:       if there exists a probe split (V1, V2) such that v1, z ∈ V1 and v2 ∈ V2 then
9:          output (V1, V2) as a probe split of G and exit;
10:      else
11:         if there exists a probe split (V1, V2) such that v1 ∈ V1 and v2, z ∈ V2 then
12:            output (V1, V2) as a probe split of G and exit;
13:         end if
14:      end if
15:   end for
16: end if
17: Output a message showing that G has no probe split.
```

1. If $(p, q) \in E$ then $(v_1, q) \notin E$ or $(p, v_2) \notin E$, and
2. if $(p, q) \notin E$ then $(v_1, q) \in E$ and $(p, v_2) \in E$.

In other words, $P(v_1, v_2, p, q)$ holds if the following statement does *not* hold.

$$(p, q) \in E \iff (v_1, q) \in E \quad \text{and} \quad (p, v_2) \in E$$

For $S \subseteq V$ we let $\overline{S}$ denote $V(G) - S$. Before proving the correctness of Algorithm 4, we present the following lemma.

Lemma 8.33. *Let $G = (V = \mathbb{P} + \mathbb{N}, E)$ be a connected, partitioned graph, let $S \subset V$ such that $|S|, |\overline{S}| \geq 2$, and let $v_1 \in \mathbb{P}(S)$, $v_2 \in \mathbb{P}(\overline{S})$, and $(v_1, v_2) \in E$. Then $(S, \overline{S})$ is a probe split of G if and only if the following condition holds:*
$(*)$ *There do not exist $p \in \mathbb{P}(S)$, $q \in \overline{S}$ such that $P(v_1, v_2, p, q)$ holds, nor $p \in S$, $q \in \mathbb{P}(\overline{S})$ such that $P(v_1, v_2, p, q)$ holds.*

Proof. If $(S, \overline{S})$ is a probe split, it is obvious from Proposition 8.28 that $(*)$ holds. Suppose that $(*)$ holds, but that $(S, \overline{S})$ is not a probe split. By symmetry we may assume that there exist $(a, b), (c, d) \in E(S, \overline{S})$, $a \in \mathbb{P}(S)$, $c \in S$, and $b, d \in \overline{S}$, such that $(a, d) \notin E$. Note that at least one of c and d must be a probe. Since neither $P(v_1, v_2, a, b)$ nor $P(v_1, v_2, c, d)$ holds, $(a, v_2), (v_1, b), (c, v_2), (v_1, d) \in E$. Since $P(v_1, v_2, a, d)$ does not hold, at least one of $(v_1, d), (a, v_2)$ is not an edge of G, a contradiction. □

Algorithm 4 Test whether there exists a probe split (V_1, V_2) of a partitioned probe G such that $v_1, z \in V_1$, $v_2 \in V_2$, $(v_1, v_2) \in E(G)$ and $v_1, v_2 \in \mathbb{P}(G)$.

Input: A partitioned probe graph G, $(v_1, v_2) \in E(G)$, $v_1, v_2 \in \mathbb{P}(G)$ and $z \in V(G) - \{v_1, v_2\}$.

Output: A probe split (V_1, V_2) of G such that $v_1, z \in V_1$, $v_2 \in V_2$ if any exists.

```
1: S ← {v1, z}; S* ← {z};
2: while S* ≠ ∅ do
3:    Select p ∈ S*;
4:    S* ← S* − {p};
5:    if p ∈ ℙ(G) then
6:       for q ∈ V(G) − S do
7:          if P(v1, v2, p, q) holds then
8:             S ← S + q; S* ← S* + q;
9:          end if
10:      end for
11:   else
12:      for q ∈ ℙ_G(V(G) − S) do
13:         if P(v1, v2, p, q) holds then
14:            S ← S + q; S* ← S* + q;
15:         end if
16:      end for
17:   end if
18: end while
19: if |S| < |V(G)| − 1 then
20:    Output (S, V(G) − S) as a probe split of G;
21: else
22:    Output the message that G has no such probe split.
23: end if
```

We prove the correctness of Algorithm 4 by induction. Initially we assume that there exists a probe split (V_1, V_2) such that $S = \{v_1, z\} \subseteq V_1$ and $v_2 \in V_2$. By the induction hypothesis, $\{v_1, z\} \subseteq S \subseteq V_1$ and $v_2 \in V_2$. If we discover $p \in \mathbb{P}(S)$, $q \in \overline{S}$ or $p \in S$, $q \in \mathbb{P}(\overline{S})$ such that $P(v_1, v_2, p, q)$ holds, then we know that $q \in V_1$, since $v_1, p \in V_1$ and $v_2 \in V_2$ are fixed. It must be that $S + q \subseteq V_1$. On the other hand, if no such p, q exist and $|S| < |V(G)| - 1$, then $(S, \overline{S})$ is a probe split. We have proved the following lemma.

Lemma 8.34. *If Algorithm 4 terminates with* $|S| < |V(G)| - 1$*, then* $(S, \overline{S})$ *is a probe split of* G*. Otherwise* G *has no probe split* (V_1, V_2) *with* $v_1, z \in V_1$ *and* $v_2 \in V_2$*.*

It is easy to verify that Algorithm 3 can be implemented in $O(n^3)$ time. Thus we have the following theorem.

Theorem 8.35. *Recognition of a partitioned probe distance-hereditary graph can be done in* $O(n^4)$ *time. If* G *is partitioned probe distance hereditary, an embedding of* G *can be computed in* $O(n^4)$.

By the discussion above, we obtain the following characterization of probe distance-hereditary graphs.

Theorem 8.36. *Let* $G = (\mathbb{P} + \mathbb{N}, E) = (\mathbb{P}', \mathbb{N}', \mathbb{P}'', \mathbb{N}'', E)$ *denote a partitioned probe graph, with the further partition* $\mathbb{P} = \mathbb{P}' + \mathbb{P}''$ *and* $\mathbb{N} = \mathbb{N}' + \mathbb{N}''$. *Then every partitioned probe distance-hereditary graph* $G = (\mathbb{P} + \mathbb{N}, E)$ *can be obtained as follows.*

1. *The following graphs are partitioned probe distance hereditary:*

$$(\{x\}, \varnothing, \varnothing, \varnothing, \varnothing) \quad \text{and} \quad (\varnothing, \{x\}, \varnothing, \varnothing, \varnothing).$$

2. *Let* $G_i = (\mathbb{P}_i', \mathbb{N}_i', \mathbb{P}_i'', \mathbb{N}_i'', E_i)$, $i = 1, 2$, *be partitioned and probe distance hereditary, and let*

$$E^* = \{(x,y) \mid x \in \mathbb{P}_1' \text{ and } y \in \mathbb{P}_2'\} \cup \{(x,y) \mid x \in \mathbb{P}_1' \text{ and } y \in \mathbb{N}_2'\} \cup \{(x,y) \mid x \in \mathbb{N}_1' \text{ and } y \in \mathbb{P}_2'\}.$$

   Then the following are also partitioned probe distance-hereditary graphs.
   (a) $(\mathbb{P}_1' + \mathbb{P}_2', \mathbb{N}_1' + \mathbb{N}_2', \mathbb{P}_1'' + \mathbb{P}_2'', \mathbb{N}_1'' + \mathbb{N}_2'', E_1 + E_2)$,
   (b) $(\mathbb{P}_1' + \mathbb{P}_2', \mathbb{N}_1' + \mathbb{N}_2', \mathbb{P}_1'' + \mathbb{P}_2'', \mathbb{N}_1'' + \mathbb{N}_2'', E_1 + E_2 + E^*)$, *and*
   (c) $(\mathbb{P}_1', \mathbb{N}_1', \mathbb{P}_1'' + \mathbb{P}_2'' + \mathbb{P}_2', \mathbb{N}_1'' + \mathbb{N}_2'' + \mathbb{N}_2', E_1 + E_2 + E^*)$.
3. *There are no other partitioned probe distance-hereditary graphs.*

For the *cliquewidth* parameter of graphs we refer to [55]. In case a cliquewidth expression for a graph with bounded cliquewidth is available, many NP-complete problems become tractable in polynomial time [55]. Unfortunately, obtaining a cliquewidth expression for graphs with cliquewidth at most 4 is still an open problem. It was shown that the cliquewidth of a distance-hereditary graph is at most 3, [56]. From Theorem 8.36, we easily obtain the following corollary.

Corollary 8.37. *If* G *is probe distance hereditary then the cliquewidth of* G *is at most 5.*

References

1. Anstee, R. P., Properties of $(0,1)$-matrices with no triangles, *Journal of Combinatorial Theory, Series A* **29** (1980), pp. 186–198.
2. Anstee, R. P. and Martin Farber, Characterizations of totally balanced matrices, *Journal of Algorithms* **5** (1984), pp. 215–230.
3. Arikati, S. and C. Rangan, An efficient algorithm for finding a two–pair, and its applications, *Discrete Applied Mathematics* **31** (1991), pp. 71–74.
4. Ausiello, G., A. D'Atri, and M. Moscarini, Chordality properties on graphs and minimal conceptual connections in semantic data models, *J. Comput. System Sci.* **33** (1986), pp. 179–202.
5. Bacsó and Z. Tuza, Dominating cliques in P_5–free graphs, *Period. Math. Hung.* **21** (1990), pp. 303–308.
6. Bandelt, H. J. and H. M. Mulder, Distance-hereditary graphs, *Journal of Combinatorial Theory, Series B* **41** (1986), pp. 182–208.
7. Berge, C., Balanced matrices, *Math. Programming* **2** (1972), pp. 19–31.
8. Berge, C., *Graphs and Hypergraphs*, North-Holland, Amsterdam, 1973.
9. Berge, C., Motivations and history of some of my conjectures, *Discrete Mathematics* **165/166** (1997), pp. 61–70.
10. Berge, C. and C. Chvatal ed., *Topics on Perfect Graphs*, Annals of Discrete Mathematics **21**, 1984.
11. Berry, Anne, Jean–Paul Bordat, and Pinar Heggernes, Recognizing weakly chordal graphs by edge separability, *Nordic Journal of Computing* **7** (2000), pp. 164–177.
12. Berry, A., J.-P. Bordat, P. Heggernes, G. Simonet, and Y. Villanger, A wide-range algorithm for minimal triangulation from an arbitrary ordering. Technical Report, LIMOS/RR-03-02, 2003.
13. Berry, Anne, M. C. Golumbic, and M. Lipshteyn, Recognizing and triangulating chordal probe graphs. Technical Report, LIMOS/RR-03-8, 2003.
14. Berry, Anne, M. C. Golumbic, and M. Lipshteyn, Two tricks to triangulate chordal probe graphs in polynomial time, *Proceedings of the 15$^{\text{th}}$ ACM–SIAM Symposium on Discrete Algorithms* (2004), pp. 962–969.
15. Bertossi, A. A., Dominating sets for split and bipartite graphs. *Information Processing Letters* **19** (1984), pp. 37-40.
16. Bhogle, S., Claude Berge, *Current Science* **83** (2002), pp. 906–907.

17. Bienstock, D., On the complexity of testing for odd holes and induced odd paths, *Discrete Mathematics* **90**, (1991), pp. 85–92.
Corrigendum in: *Discrete Mathematics* **102** (1992), pp. 109.
18. Blair, J. R. S. and B. Peyton, An introduction to chordal graphs and clique trees. In: (A. George *et al.* eds.) *Graph theory and sparse matrix computation* Springer, New York, 1993, pp. 1–29.
19. Bodlaender, H. L., A linear time algorithm for finding tree decompositions of small treewidth, *SIAM J. Comput.* **25** (1996), pp. 1305–1317.
20. H. Bodlaender, T. Kloks and D. Kratsch, Treewidth and pathwidth of permutation graphs, *SIAM Journal on Discrete Mathematics* **8** (1995), pp. 606–616.
21. Boliac, R. and V. V. Lozin, On teh clique-width of graphs in hereditary classes, *Proceedings ISAAC'02* LNCS 2518 (2002), pp. 44–54.
22. Bonomo, Flavia, Guillermo Durán, Min Chih Lin, and Jayme L. Szwarcfiter, On balanced graphs, *Mathematical Programming* **105** (2006), pp. 233–250.
23. K. S. Booth and G. S. Lueker, Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms, *Journal of Computer and System Sciences* **13** (1976), pp. 335–379.
24. Vincent Bouchitte and Ioan Todinca, Listing all potential maximal cliques of a graph, *Theoretical Computer Science* **276** (2002), pp. 17–32.
25. Brandstädt, A., Classes of bipartite graphs related to chordal graphs, *Discrete Applied Mathematics* **32** (1991), pp. 51–60.
26. Brändstadt, A., V. B. Le, and J. P. Spinrad, *Graph classes: A survey* SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 1999.
27. Bretscher, A., D. Corneil, M. Habib, and C. Paul, A simple linear time LexBFS cograph recognition algorithm, *Proceedings WG'03*, LNCS 2880 (2003), pp. 119–130.
28. Brouwer, A.E., P. Duchet, A. Schrijver, Graphs whose neighborhoods have no special cycles, *Discrete Mathematics* **47** (1983), pp. 177–182.
29. Buneman, P., A characterization of rigid circuit graphs, *Discrete Mathematics* **9** (1974), pp. 205–212.
30. Bürgisser, Peter, Michael Clausen, and Mohammed Amin Shokrollahi, *Algebraic complexity theory*, Springer Verlag, Berlin 1997.
31. Burlet, M. and J. Fonlupt, Polynomial algorithm to recognize a Meyniel graph. In (C. Berge and V. Chvátal, eds.) *Topics on perfect graphs*, Annals of Discrete Mathematics, Vol. 21, North-Holland, Amsterdam, 1984.
32. Cameron, K. and J. Edmonds, Existentially polytime theorems. DIMACS, Ser. Discrete Math. Theor. Comput. Sci. **1** (1990), pp. 83–100.
33. Capelle, Christian, Alain Cournier, and Michel Habib, Cograph recognition algorithm revisited and online induced P_4–search. Rapport technique 94073, LIRMM, 1994.
34. Chang, G. J., k-*domination and graph covering problems*, PhD thesis, School of OR and IE, Cornell University, Ithica, NY, 1982.
35. Chang, G. J., and T. Kloks, On the number of minimal clique separators in a graph. Manuscript 2004.
36. Chang, G. J., A. J. J. Kloks, J. Liu, and S.-L. Peng, The PIGs full monty - a floor show of minimal separators, *Proceedings STACS'05*, LNCS 3404 (2005), pp. 521–532.

37. Chang, G. J., T. Kloks, and S.-L. Peng, Probe interval bigraphs, *Electronic Notes in Discrete Mathematics* **7** (2005).
38. Chang, Maw-Shang, S. Y. Hsieh, and G. H. Chen, Dynamic programming on distance-hereditary graphs, *Proceedings of ISAAC'97*, LNCS 1350 (1997), pp. 344–353.
39. Chang, M.-S., T. Kloks, D. Kratsch, J. Liu, and S.-L. Peng, On the recognition of probe graphs of some self-complementary graph classes, *Proceedings of the 11th Annual International Conference COCOON'05*, LNCS 3595, pp. 808–817.
40. Chvátal, V., On certain polytopes associated with graphs, *J. Combin. Theory B* **18** (1975), pp. 138–154.
41. Chvátal, Vašek, Irena Rusu, and R. Sritharan, Dirac-type characterizations of graphs without long chordless cycles, *Discrete Mathematics* **256** (2002), pp. 445–448.
42. Chudnovsky, M., K.–I. Kawarabayashi, and P. Seymour, Detecting even holes, *Journal of Graph Theory* **48**, (2005), pp. 85–111.
43. Chudnovsky, M., N. Robertson, P. Seymour, and R. Thomas, The strong perfect graph theorem. Preprint 2002.
http://www.math.princeton.edu/ mchudnov/perfect.pdf
44. Conforti, M. and G. Cornuéjols, Balanced $0, \pm 1$-matrices, bicoloring and total dual integrality, *Mathematical Programming* **71** (1995), pp. 249–258.
45. Conforti, M., G. Cornuéjols, X. Liu, K. Vušković, and G. Zambelli, Odd hole recognition in graphs of bounded clique size. Preprint 2004.
46. Conforti, M., G. Cornuéjols, and R. Rao, Decomposition of balanced matrices, *J. Comb. Theory B* **77** (1999), pp. 292–406.
47. Coppersmith, D. and S. Winograd, Matrix multiplication via arithmetic progressions, *Proceedings 19th ACM Syposium on Theory of Computing* (1987), pp. 1–6.
48. Cormen, Thomas H., Charles E. Leiserson, and Ronald L. Rivest, *Introduction to algorithms*, The MIT press, Cambridge, Massachusetts, 1990.
49. Corneil, D. G., M. Habib, J.-M. Lanlignel, B. Reed, and U. Rotics, Polynomial time recognition of clique-width ≤ 3 graphs, *Proceedings of LATIN'2000*, LNCS 1776 (2000), pp. 126–134.
50. Corneil, D. G., H. Lerchs, and L. Stewart-Burlingham, Complement reducible graphs, *Discrete Applied Mathematics* **3** (1981), pp. 163–174.
51. Corneil, D. G., and Y. Perl, Clustering and domination in perfect graphs. *Discrete Applied Mathematics* **9** (1984), pp. 27-39.
52. Corneil, D. G., Y. Perl, and L. K. Stewart, A linear recognition algorithm for cographs, *SIAM Journal on Computing* **14** (1985), pp. 926–934.
53. Cornuéjols, G. and W. H. Cunningham, Compositions for perfect graphs, *Discrete Mathematics* **55** (1985), pp. 245–254.
54. Cornuéjols, Gérard, Xinming Liu, and Kristina Vušković, A polynomial algorithm for recognizing perfect graphs, *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, 2003.
55. Courcelle, B., The expression of graph properties and graph transformations in monadic second–order logic. In: (Grzegorz Rozenberg, ed.) *Handbook of graph grammars and computing by graph transformations. Vol. 1: Foundations*, 1997.
56. Courcelle, B. and S. Olariu, Upperbounds to the cliquewidth of graphs, *Discrete Applied Mathematics* **101** (2000), pp. 77–114.

57. Cowan, D. D., L. O. James, and R. G. Stanton, Graph decomposition for undirected graphs, *3rd South-Eastern Conference on Combinatorics, Graph Theory, and Computing*, Utilitas Math. (1972), pp. 281–290.
58. Cunningham, W. H., Decomposition of directed graphs, *SIAM J. Algebraic Discrete Methods* **3** (1982), pp. 214–228.
59. D'Atri, A. and M. Moscarini, Distance-hereditary graphs, Steiner trees, and connected domination, *SIAM J. Comput.* **17** (1988), pp. 521–538.
60. Dahlhaus, E., *Chordale Graphen im besonderen Hinblick auf parallele Algorithmen*, Habilitation thesis, Universität Bonn, 1991.
61. Dahlhaus, E., Efficient parallel recognition algorithms for cographs and distance–hereditary graphs, *Discrete Applied Mathematics* **57** (1995), pp. 29–44.
62. Dahlhaus, E., J. Gustedt, and R. M. McConnell, Efficient and practical modular decomposition, *Proceedings 8th ACM–SIAM Symposium on Discrete Algorithms* (1997), pp. 26–35.
63. Dahlhaus, Elias, Paul D. Manuel, and Mirka Miller, Maximum h-colorable subgraph problem in balanced graphs, *Information Processing Letters* **65** (1998), pp. 301–303.
64. Damiand, G., M. Habib, and C. Paul, A simple paradigm for graph recognition: Application to cographs and distance-hereditary graphs, *Theoretical Computer Science* **263** (2001), pp. 99–111.
65. Dantzig, G. B., "Application of the simplex method to a transportation problem," in *Activity analysis of production and allocation* (T. C. Koopmans, ed.), Wiley, New York, 1951.
66. Dirac, G., On rigid circuit graphs, *Abh. Math. Sem. Univ. Hamburg* **25** (1961), pp. 71–76.
67. Dragan, F. F. and F. Nicolai, LexBFS-orderings of distance-hereditary graphs, Schriftenreihe des Fachsbereichs Mathematik der Universität Duisburg, Duisburg, Germany, SM–DU–303 (1995).
68. Dragan, F. F., F. Nicolai, and A. Brandstädt, Convexity and HHD-free graphs, *SIAM J. Discrete Math.* **12** (1999), pp. 119–135.
69. Dushnik, B. and E. W. Miller, Partially ordered sets, *Amer. J. Math.* **63** (1941), pp. 600–610.
70. Even, S., A. Pnueli, and A. Lempel, Permutation graphs and transitive graphs, *J. Assoc. Comput. Mach.* **19** (1972), pp. 400–410.
71. Farber, M., Characterizations of strongly chordal graphs, *Discrete Mathematics* **43** (1983), 173–189.
72. Farber, M. and R. E. Jamison, Convexity in graphs and hypergraphs, *SIAM J. Alg. Discrete Methods* **7** (1986), pp. 433–444.
73. Földes, S. and P. L. Hammer, Split graphs, *Congressus Numerantium* **19** (1977), pp. 311–315.
74. Fulkerson, D. R. and O. A. Gross, Incidence matrices and interval graphs, *Pacific Journal of Math.* **15** (1965), pp. 835–855.
75. Fulkerson, D. R., A. J. Hoffman, and R. Oppenheim, On balanced matrices, *math. Programming* **1** (1974), pp. 120–132.
76. T. Gallai, Transitiv orientierbare Graphen, *Acta Math. Sci. Hung.* **18** (1967), pp. 25–66.
77. Galinier, P., M. Habib, and C. Paul, Graphs and their clique graphs, *Proceedings WG'95* LNCS 1017 (1995), pp. 358–371.

78. Gasparyan, G. S., minimal imperfect graphs: A simple approach, *Combinatorica* **16** (1996), pp. 209–212.
79. Gavril, F., The intersection graphs of subtrees in trees are exactly the chordal graphs, *Journal of Combinatorial Theory, Series B* **16** (1974), pp. 47–56.
80. Ghouila-Houri, A., Caractérisation des matrices totallement unimodulaires, *C. R. Acad. Sc. Paris* **254** (1962), pp. 1192–1194.
81. Goh, L. and D. Rotem, Recognition of perfect elimination bipartite graphs, *Information Processing Letters* **15** (1982), pp. 179–182.
82. Goldberg, P. W., M. C. Golumbic, H. Kaplan, and R. Shamir, Four strikes against physical mapping of DNA, *J. Comput. Biol.* **2** (1995), pp. 139–152.
83. Golumbic, M. C., The complexity of comparability graph recognition and coloring, *Computing* **18** (1977), pp. 199–208.
84. Golumbic, M. C., The complexity of comparability graph recognition and coloring, *J. Combin. Theory Ser. B* **22** (1977), pp. 68–90.
85. Golumbic, M. C., Trivially perfect graphs, *Discrete Mathematics* **24** (1978), pp. 105–107.
86. Golumbic, M. C., *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.
87. Golumbic, M. C. and C. F. Goss, Perfect elimination and chordal bipartite graphs, *Journal of Graph Theory* **2** (1978), pp. 155–163.
88. Golumbic, M. C., H. Kaplan, and R. Shamir, Graph sandwich problems, *Journal of Algorithms* **19** (1995), pp. 449–473.
89. Golumbic, M. C. and M. Lipshteyn, Chordal probe graphs, *Discrete Applied Mathematics* **143** (2004), pp. 221–237.
90. Golumbic, M. C. and U. Rotics, On the clique-width of some perfect graph classes, *International Journal of Foundations of Computer Science* **11** (2000), pp. 423–443.
91. Golumbic, M. C. and R. Shamir, Complexity and algorithms for reasoning about time: a graph theoretic approach, *J. Assoc. Comput. Mach.* **40** (1993), pp. 1108–1133.
92. Golumbic, M. C. and Ann N. Trenk, *Tolerance Graphs*, Cambridge studies in advanced mathematics 89, 2004.
93. Gross, Jonathan L. and Jay Yellen, *Handbook of graph theory and applications*, CRC Press, 2003.
94. Grötschel, Martin, Characterizations of perfect graphs, *Mathematical Programming Society Newsletter* **62**, (1999).
95. Grötschel, M., L. Lovász, and A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* **1** (1981), pp. 169–197. Corrigendum: *Combinatorica* **4** (1984), pp. 291–295.
96. Grötschel, M., L. Lovász, and A. Schrijver, Polynomial algorithms for perfect graphs, *Annals of Discrete Mathematics* **21** (1984), pp. 325–356.
97. Haas, R. and M. Hoffmann, Chordless paths through three vertices, *Proceedings of the Parameterized and Exact Computation First International Workshop, IWPEC'04*, LNCS 3162 (2004), pp. 25–36.
98. Habib, Michel, Fabien de Montgolfier, and Christophe Paul, A simple linear-time modular decomposition algorithm for graphs, using order extensions, *Proceedings SWAT'04*, LNCS 3111 (2004), pp. 187–198.
99. Habib, M. and C. Paul, A simple linear time algorithm for cograph recognition, *Discrete Applied Mathematics* **145** (2005), pp. 183–197.

100. Hajnal, A. and J. Surányi, Über die Auflösing von Graphen in vollständige Teilgraphen, *Ann. Univ. Sci. Budapest, Eötvös Sect. Math.* **1** (1958), pp. 113–136.
101. Hammer, P. L. and F. Maffray, Completely separable graphs, *Discrete Applied Mathematics* **27** (1990), pp. 85–99.
102. Hammer, P. L., F. Maffray, and M. Preissmann, A characterization of chordal bipartite graphs. RUTCOR Research Report, Rutgers University, New Brunswick, NJ, RRR, 1989, pp. 16–89.
103. Hammer, P. L. and B. Simeone, The splittance of a graph, *Combinatorica* **1** (1981), pp. 275–284.
104. Harary, F., J. A. Kabell, and F. R. McMorris, Interval bigraphs, *Comment. Math. Univ. Carolinae* **23** (1982), pp. 739–745.
105. Hayward, R. B., Weakly triangulated graphs, *Journal of Combinatorial Theory, series B* **39** (1985), pp. 200–208.
106. Hayward, R. B., Meyniel weakly triangulated graphs–I: co–perfect orderability, *Discrete Applied Mathematics* **73** (1997), pp. 199–210.
107. Hayward, Ryan and Bruce A. Reed, "Forbidding holes and antiholes," in *Perfect graphs* (Jorge L. Alfonsín Ramírez and B. A. Reed, eds.), Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, Chichester, 2001.
108. Hayward, R. B., J. S. Spinrad, and R. Sritharan, Weakly chordal graph algorithms via handles, *Proceedings 11th ACM–SIAM Symposium on Discrete Algorithms* (2000), pp. 42–49.
109. Hayward, H. B., C. Hoàng, and F. Maffray, Optimizing weakly chordal graphs, *Graphs and Combinatorics* **5**, ((1989), pp. 339–349. Erratum: *Graphs and Combinatorics* **6** (1990), pp. 33–35.
110. Hertz, A., Slim graphs, *Graphs and Combinatorics* **5** (1989), pp. 149–157.
111. Hoàng, C. T., *Perfect graphs*, PhD thesis, School of Computer Science, McGill University, Montreal, 1985.
112. Hoàng, C. T. and N. Khouzam, On brittle graphs, *Journal Graph Theory* **12**, (1988), 391–404.
113. Chính T. Hoàng and Frédéric Maffray, On slim graphs, even pairs, and star-cutsets, *Discrete Mathematics* **105** (1992), pp. 93–102.
114. Hoàng, Chín T., Frédéric Maffray, Stephan Olariu, and Myram Preissmann, A charming class of perfectly orderable graphs, *Discrete Mathematics* **102** (1992), pp. 67–74.
115. Hoàng, C. T. and R. Sritharan, Finding houses and holes in graphs, *Theoretical Computer Science* **259**, (2001), pp. 233–244.
116. Hoffman, A. J., A. W. J. Kolen, and M. Sakarovitch, Totally-balanced and greedy matrices, *SIAM J. Alg. Discrete Methods* **6** (1985), pp. 721–730.
117. Hoffman, A. J. and J. B. Kruskal, "Integral boundary points of convex polyhedra," in *Linear inequalities and related systems* (H. Kuhn and A. Tucker, eds.), Princeton University Press, Princeton, 1958.
118. Howorka, E., A characterization of distance-hereditary graphs, *The Quarterly Journal of Mathematics* **28** (1977), pp. 417–420.
119. Howorka, E., A characterization of ptolemaic graphs, *Journal of Graph Theory* **5** (1981), pp. 323–331.
120. Hshieh, S.-Y., C.-W. Ho, T.-S. Hsu, M.-T. Ko, and G.-H. Chen, Characterization of efficiently solvable problems on distance-hereditary graphs, *Proceedings of ISAAC'98*, LNCS 1533 (1998), pp 257–266.

121. Hsieh Sun-Yuan, Chin-Wen Ho, Tsan-Sheng Hsu, Ming-Tat Ko, and Gen-Huey Chen, Efficient parallel algorithms on distance-hereditary graphs, *Parallel Processing Letters*, vol.9 (1999), pp. 43-52, 1999.
122. Hsieh, S.-Y., C.-W. Ho, T.-S. Hsu, M.-T. Ko, and G.-H. Chen, A faster implementation of a parallel tree contraction scheme and its application on distance-hereditary graphs, *Journal of Algorithms* **35** (2000), pp. 50-81.
123. Iijama, K. and Y. Shibata, A bipartite representation of a triangulated graph and its chordality, ICS 79-1, Dept. Of Comp. Sci., Genma University, 1979.
124. Jamison, B. and S. Olariu, On the semi-perfect elimination, *Advances in Applied Mathematics* **9** (1988), pp. 364–376.
125. Jamison, B. and S. Olariu, P_4-reducible graphs—a class of uniquely tree representable graphs, *Studies in Appl. Math.* **81** (1989), pp. 79–87.
126. Jamison, B. and S. Olariu, A unique tree representation for P_4-sparse graphs, *Discrete Applied Mathematics* **35** (1992), pp. 115–129.
127. Jamison, B. and S. Olariu, Recognizing P_4-sparse graphs in linear time, *SIAM Journal on Computing* **21** (1992), pp. 381–406.
128. Jamison, B. and S. Olariu, A linear time recognition algorithm for P_4–reducible graphs, *Theoretical Computer Science* **145** (1995), pp. 329–344.
129. Johnson, J. L. and J. Spinrad, A polynomial-time recognition algorithm for probe interval graphs, *Proceedings 12*th *ACM–SIAM Symposium on Discrete Algorithms* (2001), pp. 477–486.
130. Kay, D. C. and G. Chartrand, A characterization of certain ptolemaic graphs, *Canad. J. Math.* **17** (1965), pp. 342–346.
131. D. Kelly, Comparability graphs, in *Graphs and Orders*, (ed. I. Rival), D. Reidel Pub. Comp., 1985, pp. 3–40.
132. Kloks, T., *Treewidth - computations and approximations*, LNCS 842, 1994.
133. Kloks, Ton and Dieter Kratsch, Computing a perfect edge without vertex elimination ordering of a chordal bipartite graph, *Information Processing Letters* **55** (1995), pp. 11–16.
134. Ton Kloks and Dieter Kratsch, Listing all minimal separators of a graph, *SIAM Journal on Computing* **27** (1998), pp. 605–613.
135. Kratsch, Dieter, A linear-time certifying algorithm to recognize split graphs. Unpublished manuscript 2005.
136. Kratsch, Dieter and Jeremy Spinrad, Between $O(nm)$ and $O(n^{\alpha})$, *Proceedings 14*th *ACM–SIAM Symposium on Discrete Algorithms* (2003), pp. 709–716.
137. Leeuwen, J. van, Graph algorithms. In: J. van Leeuwen (ed.) *Handbook of Theoretical Computer Science, A: Algorithms and Complexity*, Elsevier Science Publ., Amsterdam, 1990.
138. Lekkerkerker, C. and D. Boland, Representations of a finite graph by a set of intervals on the real line, *Fund. Math.* **51** (1962), pp. 45–64.
139. Lerchs, H., On cliques and kernels. Technical Report, Department of Computer Science, University of Toronto, 1971.
140. Lin, R. and S. Olariu, An NC recognition algorithm for cographs, *J. Parallel Dist. Comput.* **13** (1991), pp. 76–90.
141. Lovász, L., A characterization of perfect graphs, *Journal of Combinatorial Theory Series B* **13** (1972), pp. 95–98.
142. Lovász, L., *Combinatorial problems and exercises*, North-Holland, Amsterdam, 1979.

143. Lubiw, Anna, Doubly lexical orderings of matrices, *SIAM J. Computing* **16** (1987), pp. 854–879.
144. Ma, T.-H. and Spinrad J., An $O(n^2)$ algorithm for undirected split decomposition, *Journal of Algorithms* **16** (1994), pp. 145–160.
145. Maffray, F. and M. Preissmann, A translation of Gallai's paper: "Transitiv orientierbare Graphen," in Chapter 3 of (Jorge L. Ramírez Alfonsín and Bruce Reed eds.) *Perfect Graphs*, John Wiley & Sons, LTD, 2001.
146. McConnell, R. M. and J. P. Spinrad, Modular decomposition and transitive orientation, *Discrete Mathematics* **201** (1999), pp. 189–241.
147. McConnell, R. M. and J. Spinrad, Construction of probe interval graphs, *Proceedings 13th ACM–SIAM Symposium on Discrete Algorithms* (2002), pp. 866–875.
148. McDiarmid, C., B. Reed, A. Schrijver, and B. Shepherd, Non–interfering network flows, *Proc. 3th Scand. Workshop Algorithm Theorey, SWAT'92*, LNCS 621 (1992), pp. 245–257.
149. McDiarmid, C., B. Reed, A. Schrijver, and B. Shepherd, Induced circuits in planar graphs, *J. Comb. Theory Ser. B*, **60** (1994), pp. 169–176.
150. McKee, T. A., How chordal graphs work, *Bulletin of the ICA* **9** (1993), pp. 27–39.
151. McKee, T. A., A new characterization of strongly chordal graphs, *Discrete Mathematics* **205** (1999), pp. 145–147.
152. McKee, T. A., Strong clique trees, neighborhood trees, and strongly chordal graphs, *Journal of Graph Theory* **33** (2000), pp. 151–160.
153. McKee, T. A., Chordal bipartite, strongly chordal, and strongly chordal bipartite graphs, *Discrete Mathematics* **260** (2003), pp. 231–238. ERRATUM in: *Discrete Mathematics* **272** (2003), p. 307.
154. McKee, T. A., Subgraph threes in graph theory, *Discrete Mathematics* **270** (2003), pp. 3–12.
155. McKee, T. A., Requiring chords in cycles, *Discrete Mathematics* **297** (2005), pp. 182–189.
156. McMorris, F.R., Chi Wang, and P. Zhang, On probe interval graphs, *Discrete Applied Mathematics* **88** (1998), pp. 315–324.
157. Mehlhorn, Kurt, *Data Structures and Algorithms 1: Sorting and Searching*, Monographs in Theoretical Computer Science. An EATCS series Vol. 1, Springer, 1984.
158. R. H. Möhring, Algorithmic aspects of comparability graphs and interval graphs, in *Graphs and Orders*, (ed. I. Rival), D. Reidel Pub. Comp., 1985, pp. 41–101.
159. R. H. Möhring and F. J. Radermacher, Substitution decomposition for discrete structures and connections with combinatorial optimization, *Annals of Discrete Mathematics* **19** (1984), pp. 257–356.
160. Müller, Haiko, Recognizing interval digraphs and interval bigraphs in polynomial time, *Discrete Applied Mathematics* **78** (1997), pp. 189–205.
161. Ngo, Hung Q. and Ding-Zhu Du, A survey of combinatorial group testing algorithms with applications to DNA library screening, *Proceedings of the DIMACS Workshop on Discrete Mathematical Problems and Medical Applications* (1999), Providence, RI, Amer. Math. Soc.
162. Nicolai, F., *Strukturelle und Algorithmische Aspekte distanz-erblicher Graphen und verwandter Klassen*, Dissertation Thesis, Gerhard Mercator Universität Duisburg, 1994.
163. Nicolai, F. and T. Szymczak, Homogeneous sets and domination: a linear time algorithm for distance-hereditary graphs, *Networks* **37** (2001), pp. 117–128.

164. Nikolopoulos, Stavros D. and Leonidas Palios, Hole and anti-hole detection in graphs, *Proceedings 15th ACM-SIAM Symposium on Discrete Algorithms, SODA'04* (2004), pp. 850–859.
165. Nikolopoulos, Stavros D. and Leonidas Palios, Recognizing HHD-free and Welsh-Powell opposition graphs, *Proceedings WG'04*, LNCS 3353 (2004), pp. 105–116.
166. Nikolopoulos, Stavros D. and Leonidas Palios, Recognizing HHDS-free graphs, *Proceedings WG'05*, LNCS 3787 (2005), pp. 456–467.
167. Padberg, Manfred, Total unimodularity and the Euler subgraph problem, *Operation Research Letters* **7** (1988), pp. 173–179.
168. Paige, R. and R. E. Tarjan, Three partition refinement algorithms, *SIAM J. Computing* **16** (1987), pp. 973–989.
169. Pelsmajer, Michael J., Jacent Tokaz, and Douglas B. West, New proofs for strongly chordal graphs and chordal bipartite graphs. Preprint August 2, 2004.
170. Pnueli, Amir, Abraham Lempel, and Shimon Even, Transitive orientation of graphs and identification of permutation graphs, *Canad. J. Math.* **23** (1971), pp. 160–175.
171. Poljak, S., A note on the stable sets and colorings of graphs, *Comment. Math. Univ. Carolin.* **15** (1974), pp. 307–309.
172. Pržulj, Nataša and Derek G. Corneil, 2-tree probe interval graphs have a large obstruction set. Manuscript 2004.
173. Rose, D. J., Triangulated graphs and the elimination process, *J. Math. Analys. Appl.* **32** (1970), pp. 597–609.
174. Rose, D. J., R. E. Tarjan, and G. S. Luecker, Algorithmic aspects of vertex elimination on graphs, *SIAM J. Comput.* **5** (1976), pp. 266–283.
175. Shannon, C. E., The zero–error capacity of a noisy channel, *I.R.E. Trans. Inform. Theory* **IT-2** (1956), pp. 8–19.
176. Sheng, L., Cycle free probe interval graphs, *Congressus Numerantium* **140** (1999), pp. 33–42.
177. Sheng, L., C. Wang, and P. Zhang, Tagged probe interval graphs. DIMACS, Rutgers Technical Report, 1998/98-12.
178. Simon, Klaus and Paul Trunz, A cleanup on transitive orientation, *Proceedings ORDAL'94*, LNCS 831 (1994), pp. 59–85.
179. Skeide, Lars Severin, *Recognizing weakly chordal graphs*, PhD thesis, Department of Informatics, University of Bergen, Norway, November, 2002.
180. Soltan, V. P., d-convexity in graphs, *Soviet Math. Dokl.* **28** (1983), pp. 419–421.
181. Spinrad J. P., Finding large holes, *Information Processing Letters* **39** (1991), pp. 227–229.
182. Spinrad, J. P., Doubly lexical ordering of dense 0-1 matrices, *Information Processing Letters* **45** (1993), pp. 229–235.
183. Spinrad, Jeremy P., *Efficient graph representations*, American Mathematical Society, 2003.
184. Spinrad J. P. and R. Sritharan, Algorithms for weakly triangulated graphs, *Discrete Applied Mathematics* **59** (1995), pp. 181–191.
185. Seymour, P., Decomposition of regular matroids, *J. Combin. Theory Ser. B* **28** (1980), pp. 305–359.
186. Tarjan, R. E. and M. Yannakakis, Simple linear time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, *SIAM J. Comput.* **13** (1984), pp. 566–579.

187. Tsukiyama, S., M. Ide, H. Ariyoshi, and I. Shirakawa, A new algorithm for generating all the maximal independent sets, *SIAM Journal on Computing* **6** (1977), pp. 505–517.
188. Tyshkevich, R. I. and A. A. Chernyak, Canonical partition of a graph defined by the degrees of its vertices, *Isv. Akad. Nauk BSSR Ser. Fiz.-Mat. Nauk* **5** (1979), pp. 14–26.
189. Uehara, Ryuhei, Canonical data structure for probe interval graphs. Manuscript 2004/2/7.
190. Walter, J. R., Representations of rigid cycle graphs, PhD thesis, Wayne State University, Detroit, 1972.
191. Wolke, E. S., The comparability graph of a tree, *Proc. Amer. Math. Soc.* **13** (1962), pp. 789–795.
192. Zhang, P, Probe interval graph and its application to physical mapping of DNA. Manuscript 1994.
193. Zhang, P, E. A. Schon, S. G. Fisher, E. Cayanis, J. Weiss, S. Kistler, and P. E. Bourne, An algorithm based on graph theory for the assembly of contigs in physical mapping of DNA, *CABIOS* **10** (1994), pp. 309–317.
194. Zhang, P., X. Ye, L. Liao, J. Russo, S. G. Fischer, Integrated mapping package - a physical mapping software tool kit, *Genomics* **55** (1999), pp. 78–87.
195. Zhu, Dingzhu, Ding-Zhu Du, and Frank K. Hwang, *Combinatorial group testing and its applications*, World Scientific Publishing Co. Inc., River Edge, NJ, 1993.

Index