RECOGNIZING CHORDAL PROBE GRAPHS AND CYCLE-BICOLORABLE GRAPHS

ANNE BERRY†, MARTIN CHARLES GOLUMBIC‡, AND MARINA LIPSHTEYN‡

Abstract. A graph \( G = (V,E) \) is a chordal probe graph if its vertices can be partitioned into two sets, \( P \) (probes) and \( N \) (non-probes), where \( N \) is a stable set and such that \( G \) can be extended to a chordal graph by adding edges between non-probes. We give several characterizations of chordal probe graphs, first, in the case of a fixed given partition of the vertices into probes and non-probes, and second, in the more general case where no partition is given. In both of these cases, our results are obtained by introducing new classes, namely, \( N \)-triangulatable graphs and cycle-bicolorable graphs. We give polynomial time recognition algorithms for each class. \( N \)-triangulatable graphs have properties similar to chordal graphs, and we characterize them using graph separators and using a vertex elimination ordering. For cycle-bicolorable graphs, which are shown to be perfect, we prove that any cycle-bicoloring of a graph renders it \( N \)-triangulatable. The corresponding recognition complexity for chordal probe graphs, given a partition of the vertices into probes and non-probes, is \( O(|P||E|) \), thus also providing an interesting tractable subcase of the chordal graph sandwich problem. If no partition is given in advance, the complexity of our recognition algorithm is \( O(|E|^2) \).

Key words. chordal graph, probe graph, triangulation, perfect graph, elimination scheme, bicoloring

AMS subject classifications. 05C17, 05C75

DOI. 10.1137/050637091

1. Introduction.

1.1. Motivation: Interval probe graphs. The study of chordal probe graphs [11, 12] was originally motivated as a generalization of the interval probe graphs which occur in applications involving physical mapping of DNA. Interval probe graphs were introduced by Zhang [20, 24] to model problems in physical mapping of DNA when the intervals are either probes or non-probes, and the information on the overlaps between non-probes is missing. As a result, Zhang defined a graph to be interval probe if its vertex set can be partitioned into probes and non-probes in such a fashion that it can be completed into an interval graph by adding only edges between non-probes. This shows two different facets of the problem: either the partition is given in advance, or a partition has to be proposed as part of the solution.

Recently, for partitioned interval probe graphs, an \( O(n^2) \) time recognition algorithm was first reported in [15] which uses PQtrees. Another method, given in [18], uses modular decomposition and has complexity \( O(n + m \log n) \) for a graph with \( n \) vertices and \( m \) edges. In the case of trees, Sheng [22] gives characterizations by a family of forbidden subgraphs for both the partitioned and non-partitioned case, thus ensuring polynomial time recognition of trees which are interval probe graphs (see also [13]). The polynomial time complexity of recognizing interval probe graphs (when no partition is given) has been given in [5].

Received by the editors July 29, 2005; accepted for publication (in revised form) January 11, 2007; published electronically July 18, 2007.


†LIMOS UMR CNRS 6158, Ensemble Scientifique des Cézeaux, Bat ISIMA D123, Université Blaise Pascal, F-63 173 Aubière, France (berry@isima.fr).

‡Caesarea Rothschild Institute and Department of Computer Science, University of Haifa, Haifa, Israel (golumbic@cs.haifa.ac.il, marina_lipstein@atricia.com).
1.2. Chordal probe graphs and their generalizations. Generalizing interval probe graphs, Golubic and Lipshteyn [11, 12] introduced chordal probe graphs as a new class of perfect graphs: A graph \( G \) is chordal probe if its vertex set can be partitioned into a set \( P \) of probes and a stable set \( N \) of non-probes in such a fashion that \( G \) can be completed into a chordal graph by adding only edges between non-probes. They gave \( O(m^2) \) algorithms to recognize chordal probe graphs which are also even-chordal, which is exactly the subfamily of chordal probe graphs which have no even hole; this class includes the interval probe graphs and is also weakly chordal [14].

Among the results in this paper, we solve the general problem for chordal probe graphs, by giving polynomial time recognition algorithms for the partitioned as well as the non-partitioned case. In doing so, we introduce two new graph superclasses, the \( N \)-triangulatable graphs and cycle-bicolorable graphs, proving interesting properties for both of them.

In Part I, we examine the partitioned case. In fact, we solve a broader problem in which the set \( N \) is not assumed to be a stable set, which defines the class of \( N \)-triangulatable graphs. We investigate the structural properties of this class and show that several properties of chordal graphs can be extended to this class, namely we characterize them using graph separators and using a vertex elimination ordering. These results enable us to propose a recognition algorithm with a complexity of \( O(|P|m) \). Section 3 deals with \( N \)-triangulatable graphs and section 4 discusses the subcase of partitioned chordal probe graphs.

In Part II, we discuss the case where no partition is given in advance. Our approach uses a lemma from [12], which remarks that in a partitioned chordal probe graph, probes and non-probes must alternate on every chordless cycle. Thus, in section 6, we again solve a broader problem by introducing the class of cycle-bicolorable graphs, a superclass of chordal probe graphs. In section 7, we characterize chordal probe graphs as cycle-bicolorable graphs in which one color defines a stable set and give a corresponding \( O(n^2m) \) time recognition algorithm. These results are based on a new graph decomposition, introduced in section 5, which groups together the cycles of the graph into so-called \( C \)-components. The polynomial time complexity relies on the theory of graph separators.

2. Background and previous results.

2.1. General definitions and properties of chordal graphs. The graphs in this work are undirected and finite. A graph is denoted \( G = (V, E) \), with \( n = |V| \) and \( m = |E| \). We let \( G_A \) denote the subgraph induced by a vertex set \( A \subset V \); similarly, \( \overline{G}_A \) denotes the subgraph induced by \( A \) in the complement \( \overline{G} \) of \( G \). A clique in a graph is a set of pairwise adjacent vertices, and a stable set in a graph is a set of pairwise non-adjacent vertices. We say that we saturate a set of vertices if we add all the edges necessary to make it a clique. In this paper, a connected component is a vertex set which induces a maximal connected subgraph.

The (open) neighborhood of a vertex \( x \) in graph \( G \) is the set \( N_G(x) = \{ y \neq x \mid xy \in E \} \); we will say that a vertex \( x \) sees another vertex \( y \) if \( xy \in E \). The closed neighborhood of \( x \) is \( N_G[x] = N_G(x) \cup \{ x \} \). We extend the notion of neighborhood to a set of vertices \( A \) by defining \( N_G(A) = \cup_{x \in A} N_G(x) - A \) and \( N_G[A] = N_G(A) \cup \{ A \} \). When there is no ambiguity as to which graph is referred to, the subscript will be omitted, i.e., \( N(x) \) will be written simply \( N(x) \). The degree of vertex \( x \) will be denoted by \( \deg_G(x) = |N(x)| \).

A chordless cycle of length \( k \) is denoted by \( C_k \) and we always assume \( k \geq 4 \). A hole is a chordless cycle of length at least five; a hole is called odd or even depending
on the parity of its length. An antihole is the complement of a hole. A graph $G$ is called perfect if every induced subgraph $G_A$ satisfies the equality $\omega(G_A) = \chi(G_A)$, where $\omega$ denotes the size of the largest clique and $\chi$ is the chromatic number. The Strong Perfect Graph Theorem \cite{6}, originally conjectured by Berge, states that a graph is perfect if and only if it contains neither an odd hole nor an odd antihole.

Chordal graphs are a well known family of perfect graphs. A graph is defined to be chordal if every cycle of length 4 or greater has a chord, that is, an edge joining two non-consecutive vertices of the cycle. Chordal graphs have important application areas including acyclic relational database schemes, facility location problems, statistical analysis, and the problem of tracing genetic mutations over an evolutionary period by constructing phylogenetic trees (see \cite{9, 19}).

It is often the case in such applications, however, that an input graph $G$ has edges missing due to incomplete data. This gives rise to the problem of adding additional edges $F$ in order to complete it into a chordal supergraph $G' = (V, E + F)$ of $G$. The edge set $F$ is said to be minimal if no proper subset defines a chordal graph when added; the resulting chordal graph is then called a minimal triangulation. When $|F|$ is required to be smallest possible, it is called a minimum triangulation.

The chordal graph sandwich problem (see \cite{4, 10, 23}) is another variation where a specified set of optional edges $E_0$ is given with the input, and the triangulation $F$ (not necessarily minimum) must satisfy $F \subseteq E_0$. Both the minimum triangulation problem and the chordal graph sandwich problem are NP-complete.

An undirected graph $G = (V, E)$ is a chordal probe graph if its vertex set can be partitioned into two subsets, $P$ (probes) and $N$ (non-probes), where $N$ is a stable set and there exists a completion $F \subseteq N \times N$ such that $H = (V, E + F)$ is a chordal graph. The class of chordal probe graphs was introduced in \cite{12} as a generalization of interval probe graphs. Interval probe graphs are defined similarly, where the completed graph $H$ must be an interval graph.

A vertex is simplicial if its neighborhood is a clique. The notion of simplicial vertex was introduced independently by Dirac in 1961 \cite{7} and by Lekkerkerker and Boland in 1962 \cite{17} as an extension of the notion of a leaf in a tree and is the basis for the following theorem by Dirac.

**Theorem 2.1** (see \cite{7}). *A non-clique chordal graph has at least two non-adjacent simplicial vertices.*

This led Fulkerson and Gross \cite{8} to characterize chordal graphs in an algorithmic manner as follows.

**Characterization 2.2** (see \cite{8}). *A graph is chordal if and only if one can repeatedly find a simplicial vertex and delete it from the graph until no vertex is left.*

This defines an ordering on the vertices called a perfect elimination ordering (peo).

One of the earliest ways which was used to compute a triangulation was to force the graph into respecting this characterization by using an ordering $\alpha$ on the vertices and repeatedly choosing the next vertex in this ordering, forcing its neighborhood into a clique by the addition of any missing edges and removing the vertex; we refer to this process as the elimination game on $(G, \alpha)$. Each of the successive graphs obtained is called a transitory elimination graph and is denoted by $G_\alpha$. At the end of the process, the set $F$ of added edges define a triangulation $G_\alpha^+ = (V, E + F)$ of the input graph $(V, E)$ (see \cite{3}).

The following property is well known.

**Property 2.3.** *If $G'$ is a triangulation of $G$ and $\alpha$ is a peo of $G'$, then $G' = G_\alpha^+$.***

**2.2. Minimal separators and minimal triangulations.** Minimal separators were introduced by Dirac \cite{7}. A subset $S$ of vertices of a connected graph $G$ is called
a separator if $G_{V-S}$ is not connected. A separator $S$ is called an ab-separator if $a$ and $b$ are in different connected components of $G_{V-S}$, a separator $S$ is a minimal ab-separator if $S$ is an ab-separator and no proper subset of $S$ is an ab-separator; finally, a separator $S$ is a minimal separator if there is some pair $\{a, b\}$ such that $S$ is a minimal ab-separator. Equivalently, a separator $S$ is minimal if there exist two distinct components $C_1$ and $C_2$ in $G_{V-S}$ such that $N'(C_1) = N'(C_2) = S$; such components are called full components of $S$.

Minimal separators turn out to be a very useful tool for computing a minimal triangulation.

**Definition 2.4** (Kloks, Kratsch, and Spinrad [16]). Let $S$ and $T$ be two minimal separators of $G$. Then $S$ crosses $T$ if there exist two components $X_1, X_2$ of $G_{V-T}$, $X_1 \neq X_2$, such that $S \cap X_1 \neq \emptyset$ and $S \cap X_2 \neq \emptyset$.

It is shown in [21] that the crossing relation is symmetric, i.e., $S$ crosses $T$ if and only if $T$ crosses $S$.

**Property 2.5.** Let $S$ and $T$ be two minimal separators of $G$. Then $S$ crosses $T$ if and only if $T$ has a vertex in each full component of $S$.

**Theorem 2.6** (see [21]). When a minimal separator $S$ is saturated, creating graph $G'$:

1. All the minimal separators which cross $S$ disappear.
2. All the minimal separators which do not cross $S$ remain.
3. No new minimal separator appears.
4. Any minimal triangulation of $G'$ is a minimal triangulation of $G$.

Thus, computing a minimal triangulation of a graph $G$ is equivalent to saturating a maximal set of pairwise non-crossing minimal separators of $G$ (see [21]).

The following is a consequence of Theorem 2.6.

**Property 2.7.** $S$ and $T$ are two crossing minimal separators of a graph $G$ if and only if $S$ contains two non-adjacent vertices $x$ and $y$ such that $T$ is a minimal xy-separator of $G$.

Lekkerkerker and Boland in [17] introduced the following notion which will be fundamental to this paper.

**Definition 2.8.** A substar $S$ of $x$ is a subset of $N(x)$ such that for some connected component $U$ of $G_{V-N[x]}$, $S = N(U)$, i.e., all the vertices of a substar see some common connected component of $G_{V-N[x]}$.

Note that the substars of $x$ are exactly the minimal separators included in the neighborhood of $x$.

**Example 2.9.** In Figure 1, the substars of $j$ are $\{b, c\}$ and $\{e, i, k\}$; $j$ is on a chordless cycle with $c, i,$ and $k$, because substar $\{e, i, k\}$ is an independent set, but $j$ is not on a cycle with $b$ nor $c$, because substar $\{b, c\}$ is a clique.

![Fig. 1. A graph. The substars of $j$ are $\{b, c\}$ and $\{e, i, k\}$.](image-url)
Additional properties which are useful are the following.

**Property 2.10** (see [1]). For a vertex \( x \), the substars of \( x \) are pairwise non-crossing.

**Property 2.11** (see [1]). Let \( x, y \) be two non-adjacent vertices of a graph \( G \), then no substar of \( x \) can cross a substar of \( y \).

**Property 2.12** (see [17]). A vertex \( x \) is on a chordless cycle if and only if at least one of its substars is not a clique. More precisely, if \( X \) is a connected component of \( G_{V−N[x]} \) such that \( S = N(X) \) contains the non-edge \( yz \), then \( x \) is on a chordless cycle \( C \) on which it sees \( y \) and \( z \), and all the other vertices of \( C \) are in \( X \).

For example, in Figure 1, vertex \( j \) is on a chordless cycle with \( e, i, \) and \( k \), because substar \( \{e, i, k\} \) is an independent set, but \( j \) is not on a cycle with \( b \) nor \( c \), because substar \( \{b, c\} \) is a clique.

This leads us to the following definition.

**Definition 2.13.** We say that a vertex \( x \) is LB-simplicial if all the substars of \( x \) are cliques.

Finally, we recall the following characterization of chordal graphs which does not appear to be well known.

**Characterization 2.14** (see [17]). A graph is chordal if and only if every vertex is LB-simplicial.

**Part I. The partitioned case.** The original motivation for this work has been the recognition of chordal probe graphs, in the non-partitioned as well as in the partitioned case. We will first address the partitioned case. In order to do this, we solve a more general problem.

3. **N-triangulatable partitioned graphs.** We introduce a new problem, namely, triangulating a graph whose vertex set is bipartitioned into “probes” and “non-probes” by adding only edges between non-probes. The corresponding class, which we call \( N \)-triangulatable graphs, is studied in this section.

One of the interesting developments is that \( N \)-triangulatable graphs turn out to be very similar to chordal graphs: We will show that several properties and characterizations of chordal graphs can be very profitably extended to \( N \)-triangulatable graphs, and that they yield the tools we need to handle this class efficiently.

**Definition 3.1.** We will say that a graph \( G = (P + N, E) \) is \( N \)-triangulatable (\( N\)-T) if a triangulation of \( G \) can be obtained by adding only edges whose endpoints are non-probes. We will call such a triangulation an \( N \)-triangulation of \( G \).

**Remark 3.2.**
1. If \( G \) is \( N \)-T, then \( G_P \) is a chordal graph.
2. An induced subgraph of an \( N \)-T graph is an \( N \)-T graph.
3. In the case where \( P = \emptyset \), the graph becomes an arbitrary graph, and it is always \( N \)-T.
4. In the case where \( N \) is a stable set, \( G \) is \( N \)-T if and only if \( G \) is chordal probe with respect to this partition.
5. Recognizing \( N \)-T graphs is a special case of the chordal graph sandwich problem, where the optional edges \( E_0 \) consist of all non-edges between non-probes.

We will now see that both Lekkerkerker and Boland’s Characterization 2.14 and Fulkerson and Gross’ Characterization 2.2 can be extended to recognize this class. These will be studied, respectively, in sections 3.1 and 3.3.

3.1. **Quasi LB-simpliciality of \( N \)-T graphs.** In this section, we extend Characterization 2.14 of chordal graphs due to Lekkerkerker and Boland to \( N \)-T graphs.
This will also enable us to give a recognition algorithm for $N$-$T$ graphs using separators.

**Definition 3.3.** We will say that a vertex $x$ is quasi-LB-simplicial if all the non-edges of all the substars of $x$ have both endpoints that are non-probes.

**Example 3.4.** In Figure 2, if black vertices are probes and white are non-probes, $c$ is quasi-LB-simplicial, as its substars are $\{b, d\}$ and $\{j\}$.

We will see that examining the substars of the probes of the graph is sufficient to characterize $N$-$T$ graphs.

**Definition 3.5.** We will say that the substars of a probe are $P$-substars.

**Theorem 3.6.** The following conditions are equivalent for a graph $G = (P + N, E)$:

1. $G$ is an $N$-$T$ graph.
2. All probes of $G$ are quasi-LB-simplicial.
3. $G$ contains no chordless cycle with two adjacent probes.

**Proof.** (1) $\Rightarrow$ (3): Let $G = (P + N, E)$ be an $N$-$T$ graph, let $V = P + N$, and let $G'$ be an $N$-triangulation of $G$. Suppose by contradiction that in $G$ there is a chordless cycle $(p_1, p_2, v_3, \ldots, v_k, p_1)$, where $p_1$ and $p_2$ are probes. In $G'$, $p_1$ sees $v_k$ and $p_2$; $v_3, v_4, \ldots, v_{k-1}$ belong to the same connected component $X$ of $G'_{V-N[p_1]}$. $N(X)$ is a substar of $G'$, but it fails to be a clique, as it contains $v_k$ and $p_2$, which are non-adjacent. This contradicts Characterization 2.14 for chordal graphs.

(3) $\Rightarrow$ (2): Assume in $G = (P + N, E)$ there is no chordless cycle with two adjacent probes. Suppose by contradiction that there exists a probe $x$ which fails to be quasi-LB-simplicial: $x$ has two non-adjacent neighbors, $y$ and $z$, one of which is a probe; w.l.o.g. $y$ is a probe. According to Property 2.12, there exists a chordless cycle which contains $y$, $x$, and $z$ consecutively, a contradiction.

(2) $\Rightarrow$ (1): Let $G = (P + N, E)$ be a graph such that all probes are quasi-LB-simplicial; we will prove that $G$ is an $N$-$T$ graph.

Let us use the minimal triangulation algorithm LB-TRIANG described in [1], which repeatedly chooses a vertex $x$, saturates its substars, and removes $x$. Regardless of the order in which the vertices are processed, LB-TRIANG computes a minimal triangulation of the input graph; we will run it by first choosing all the probes.

We claim that no new $P$-substar can appear. Because of Theorem 2.6, the only way a $P$-substar can be created is by adding edges which will cause a previous minimal separator $S$, which was not a $P$-substar, to be in the neighborhood of a probe. However, no new edge can be added incident to a probe, so this cannot happen.

After all the $P$-substars have been processed and eliminated, only vertices from $N$ are left in the graph. When we finish the execution, we will have computed a minimal
triangulation of $G$ which has added only edges between two non-probes, which is thus an $N$-triangulation of $G$. By definition, $G$ is an $N$-$T$ graph.

**Complexity.** The recognition algorithm based on Theorem 3.6 runs in $O(|P|m)$ time: The implementation of Algorithm LB-TRIANG proposed in [1], as in the proof of Theorem 3.6, uses a data structure inspired from clique trees and requires only $O(m)$ time per processed vertex; a global $O(m)$ time is then used to check that only edges between pairs of non-probes have been added.

Computing a minimal triangulation of an $N$-$T$ graph costs $O(nm)$ time, which is the same as computing a minimal triangulation of any graph. However, in order to recognize $N$-$T$ graphs, it is not necessary to actually compute an $N$-triangulation. Therefore, unless $P$ is of order $n$, it is cheaper to recognize the class than to exhibit an $N$-triangulation for it.

### 3.2. Properties of $N$-$T$ graphs.

**Theorem 3.7.** Let $G = (P + N, E)$ be an $N$-$T$ graph. The $P$-substars of $G$ are pairwise non-crossing.

**Proof.** Let $G = (P + N, E)$ be an $N$-$T$ graph, and let $V = P + N$. Let us assume by contradiction that there are two crossing $P$-substars $S_1$ and $S_2$. By Property 2.11, $S_1$ and $S_2$ must be substars of two adjacent vertices $p_1$ and $p_2$. By Property 2.7, there must be two non-adjacent vertices $x$ and $y$ in $S_1$, such that $S_2$ is a minimal $xy$-separator.

Let us first suppose that $p_2$ belongs to $S_1$. Since $p_1$ is quasi-LB-simplicial, $p_2$ must see $x$ and $y$. Therefore, every minimal $xy$-separator must contain $p_2$, which contradicts the fact that $S_2$ is a minimal $xy$-separator.

Let us now examine the case where $p_2$ is not in $S_1$. Let $X$ be the connected component of $G_{V-N}[p_1]$ such that $S_1 = \mathcal{N}(X)$. According to Property 2.12, $x$ and $y$ belong to some chordless cycle $C$ on which $x, p_1$ and $y$ are consecutive, with all other vertices in $X$. Suppose $p_2$ sees some of these intermediate vertices $C \cap (V - \mathcal{N}[p_1])$. Then $p_2$ would belong to $\mathcal{N}(X)$ and thus to $S_1$, which is impossible. Therefore, $S_2$ has no vertex in $X$, which is a full component of $S_1$; by Property 2.5, $S_1$ and $S_2$ are non-crossing.

**Corollary 3.8.** The number of $P$-substars in an $N$-$T$ graph is less than $n$.

**Proof.** This follows from the simple observation that, since the $P$-substars are non-crossing minimal separators, they can all be chosen to be saturated and preserved in some minimal triangulation of $G$, which, as all chordal graphs, has less than $n$ minimal separators.

Recall that in the proof of Theorem 3.6, we ran LB-TRIANG by using all the probes in a first phase and then the non-probes in a second phase. Since the minimal separators which are chosen as substars in the first phase are pairwise non-crossing, the resulting set $F_P$ of edges which is added is the same, regardless of the order in which the probes are processed; the edges of $F_P$ are mandatory, and we will use them to define $G^*$ below. The set of edges computed by the second phase, however, depends on the order in which the non-probes are processed.

**Definition 3.9.** We define the enhanced graph $G^*$ of $G$ to be the graph obtained from $G$ by saturating all the $P$-substars of $G$.

**Example 3.10.** Figure 3 gives the enhanced graph of the graph of Figure 2.

**Theorem 3.11.** Any minimal triangulation of $G^*$ is a minimal triangulation of $G$ and an $N$-triangulation of $G$.

**Proof.** By Theorem 3.7, the $P$-substars of $G$ are pairwise non-crossing. Therefore, $G^*$ is obtained by saturating a set of pairwise non-crossing minimal separators of $G$. 

By Theorem 2.6, any minimal triangulation of $G^*$ is a minimal triangulation of $G$. If we run Algorithm LB-TRIANG as in the proof of Theorem 3.6 by first choosing the probes, making these simplicial will only add edges between two non-probes. After the probes are processed and eliminated, only non-probes are left in the graph, and the chosen subsequent triangulation will also add only edges between two non-probes; the minimal triangulation thus obtained is an $N$-triangulation.

3.3. Quasi-perfect elimination in $N$-$T$ graphs. We will now go on to show that Fulkerson and Gross’ Characterization 2.2 can also be extended to an $N$-$T$ graph and that, as is the case with chordal graphs with respect to perfect elimination orderings (peos), a greedy approach to playing the quasi-peo elimination game will successfully recognize $N$-$T$ graphs.

Definition 3.12. Let $G = (P + N, E)$ be an $N$-$T$ graph. We will say that a vertex $v$ of $G$ is quasi-simplicial if every non-edge of $\mathcal{N}(v)$ has both endpoints which are non-probes.

Definition 3.13. We will say that an ordering $\alpha$ on the vertices of $G$ is a quasi-perfect elimination ordering (qpeo) if at each step $i$ of the elimination game on $(G, \alpha)$, vertex $\alpha(i)$ is quasi-simplicial in the transitory elimination graph $G_i$.

Example 3.14. In Figure 2, if black vertices are probes and white are non-probes, $a$ is quasi-simplicial and $d$ is not. However, if $a$ is chosen first in a qpeo, saturating $\mathcal{N}(a)$ and removing $a$ will make $d$ quasi-simplicial in the transitory graph; $\alpha = (a,d,e,b,j,h,l,f,e,k,g,i)$ is a qpeo.

Lemma 3.15. Let $G = (P + N, E)$ be an $N$-$T$ graph, and let $v$ be a quasi-simplicial vertex of $G$. If $G'$ is the graph obtained by making $v$ simplicial and removing it, then $G'$ is also $N$-$T$.

Proof. Suppose by contradiction that $G'$ fails to be an $N$-$T$ graph, we will prove that $G$ is not $N$-$T$. According to Theorem 3.6, there must be a chordless cycle $C'$ in $G'$ containing two consecutive probes. Let $X'$ be the vertex set corresponding to $C$.

If $G_{X'}$ is also a chordless cycle (in $G$), then $G$ fails to be $N$-$T$, by Theorem 3.6.

Otherwise, in cycle $G_{X'}$, there exists a unique edge $xy$ which was added while making $v$ simplicial. (If several edges were added, $C'$ would not be chordless). Let $X = X' \cup \{v\}$. Clearly, $G_X$ is a cycle, call it $C$; suppose it fails to be chordless: $v$ has a neighbor $w$ on $C$, $w \neq x,y$; but in that case, edges $xw$ and $yw$ would have been added to $G'$, which contradicts the fact that $C'$ is chordless in $G$. Thus $C$ is a chordless cycle with two consecutive probes, so by Theorem 3.6, $G$ is not $N$-$T$.

Theorem 3.16. Let $G = (P + N, E)$ be a graph. The following are equivalent:
1. \( G \) is an \( N \)-\( T \) graph.
2. \( G \) has a quasi-perfect elimination ordering.
3. A greedy elimination game on quasi-simplicial vertices succeeds.

**Proof.** (2) \( \Rightarrow \) (1): Let \( G = (P + N, E) \) be a graph with a qpeo \( \alpha \). Running the elimination game on \( (G, \alpha) \) will add only edges between two non-probes, so it will produce an \( N \)-triangulation of \( G \).

(1) \( \Rightarrow \) (2) Let \( G = (P + N, E) \) be an \( N \)-\( T \) graph, let \( G' \) be an \( N \)-triangulation of \( G \), and let \( \beta = (v_1, \ldots, v_n) \) be a peo of \( G' \). We claim that \( \beta \) is a qpeo of \( G \). By Property 2.3, \( G' = G'_N + \). Since \( v_i \) is simplicial in \( G'_{\{v_i,\ldots,v_n\}} \), it is quasi-simplicial in \( G_{\{v_i,\ldots,v_n\}} \), because the elimination game only adds edges whose endpoints are non-probes at each step. Therefore, \( \beta \) is a qpeo of \( G \).

(1) \( \Rightarrow \) (3): If the elimination game fails at some step, then the corresponding transitory graph has no quasi-simplicial vertex, so by the equivalence (1) \( \iff \) (2), it fails to be \( N \)-\( T \). Therefore, by Lemma 3.15, \( G \) is not \( N \)-\( T \).

(3) \( \Rightarrow \) (2): Trivial.

**Complexity.** A recognition algorithm can be given based on condition (3) of Theorem 3.16. This runs in \( O(n^2m') \) time, where \( m' \) is the number of edges of the \( N \)-triangulation computed by the elimination game run on a qpeo, since a brute force approach will require \( O(nm') \) time to find a quasi-simplicial vertex and process it. (A referee has pointed out that the complexity of recognizing \( N \)-\( T \) graphs in this way can be further reduced to \( O(|P|m') \).)

In any case, this complexity is not as good as the \( O(|P|m) \) time we found in section 3.1. However, there may be, as is the case for chordal graphs, a LEX M-type algorithm which could compute a qpeo in \( O(|N|m) \) time—a question we leave open.

We now will use our results to extend Dirac’s Theorem 2.1 to \( N \)-\( T \) graphs.

**Theorem 3.17.** Let \( G = (P + N, E) \) be an \( N \)-\( T \) graph which is not a clique; then in \( G \) there are at least two non-adjacent quasi-simplicial vertices.

**Proof.** By induction on the number of vertices. Clearly, any \( N \)-\( T \) graph on 4 vertices which is not a clique has two non-adjacent vertices which are quasi-simplicial.

Let us consider an \( N \)-\( T \) graph \( G \) with \( n \) vertices. By Theorem 3.16, \( G \) has a quasi-simplicial vertex \( x \). If \( x \) sees all the other vertices in \( G \), let \( G' \) be obtained by simply removing \( x \) from \( G \). By the induction hypothesis, \( G' \) has two non-adjacent quasi-simplicial vertices, which are trivially also quasi-simplicial and non-adjacent in \( G \).

Otherwise, let \( G' \) be obtained by saturating \( A_G(x) \) and removing \( x \). According to Lemma 3.15, \( G' \) is an \( N \)-\( T \) graph. By the induction hypothesis, \( G' \) must have two non-adjacent quasi-simplicial vertices, at least one of which, call it \( z \), is not in \( A_G(x) \), since in \( G' \), \( A_G(x) \) is a clique. We claim that \( z \) is quasi-simplicial in \( G \). Suppose this is not the case: In \( G \), \( z \) must see a non-edge \( \{v, w\} \), with \( v \) a probe, which is not a non-edge of \( G' \), so edge \( vw \) must have been added to \( G' \) when making \( x \) simplicial. But since \( x \) is quasi-simplicial in \( G \), there can be no such non-edge \( \{v, w\} \) in \( G \). Thus, in \( G \), \( x \) and \( z \) are two non-adjacent quasi-simplicial vertices.

4. **Recognizing partitioned chordal probe graphs.** In this section, we apply our results from section 3 on \( N \)-triangulatable graphs to characterizing and recognizing partitioned chordal probe graphs.

**Theorem 4.1.** Let \( G = (P + N, E) \), with \( N \) a stable set. The following conditions are equivalent:

1. \( G \) is chordal probe.
2. All probes of \( G \) are quasi-LB-simplicial.
3. \( G \) contains no chordless cycle with two adjacent probes.
Proof. This follows directly from Theorem 3.6 and Remark 3.2 (4).

Complexity. Provided we test that \( N \) is a stable set, the recognition algorithm for \( N\T \) graphs given in section 3.1 also recognizes chordal probe graphs, with the same \( O(|P|m) \) complexity.

**Theorem 4.2.** Let \( G = (P+N,E) \) be a graph, with \( N \) a stable set. The following three are equivalent:

1. \( G \) is a chordal probe graph.
2. \( G \) has a quasi-perfect elimination ordering.
3. A greedy elimination game on quasi-simplicial vertices succeeds.

**Proof.** This follows immediately from Theorem 3.16.

**Remark 4.3.** Theorem 4.2 defines an elimination process on neighborhoods which are split graphs, as the vertices in each neighborhood are partitioned into a clique of probes and a stable set of non-probes; moreover, they form a special kind of split graph, which can be qualified as “complete split graph,” meaning that all possible edges between a probe and a non-probe belong to the graph. This extends the simplicial elimination process on chordal graphs, where the elimination is on complete neighborhoods.

Theorem 3.17 also trivially extends to chordal probe graphs:

**Corollary 4.4.** Let \( G = (P+N,E) \) be a chordal probe graph which is not a clique; then in \( G \) there are at least two non-adjacent quasi-simplicial vertices.

**Part II. The non-partitioned case.** Having solved the partitioned case for recognizing chordal probe graphs in Part I, we will now go on to the non-partitioned case. Again, we do this by first solving a more general problem.

5. **Decomposing an arbitrary graph into \( C \)-components.** As stated in the introduction, our approach to recognizing chordal probe graphs uses a lemma from [11] which remarks that in any valid partition of a chordal probe graph, probes and non-probes must alternate on every chordless cycle. In order to study this phenomenon, we first propose a partition of the vertices of a graph into components which group together cycles of the graph, thus introducing a new graph decomposition.

**Definition 5.1.** Let \( G \) be an arbitrary graph.

1. A \( C \)-edge is an edge which belongs to some chordless cycle.
2. A \( C \)-path is a path made out of \( C \)-edges.
3. A \( C \)-component is a set of vertices in which there is a \( C \)-path connecting each pair of vertices in the component.
4. An external edge is an edge which has its endpoints in two different \( C \)-components.

**Example 5.2.** Figure 4 shows the partition into \( C \)-components of a graph.

**Property 5.3.** Let \( G \) be an arbitrary graph; being connected by a \( C \)-path is an equivalence relation on the vertices of \( G \); we will denote this relation by \( \sim \).

**Proof.** Trivially, if \( x \) and \( y \) are connected by a \( C \)-path, then \( y \) and \( x \) are also connected by the same path, thus ensuring symmetry. Transitivity: Let \( \mu_1 \) be a \( C \)-path from \( x \) to \( y \) and \( \mu_2 \) be a \( C \)-path from \( y \) to \( z \); the concatenation of \( \mu_1 \) with \( \mu_2 \) is a \( C \)-path from \( x \) to \( z \).

**Property 5.4.** Every chordless cycle is entirely contained in some \( C \)-component.

**Proof.** Two vertices belonging to some chordless cycle are connected by a \( C \)-path, so by definition they must belong to some common \( C \)-component.

**Property 5.5.** Every antihole is entirely contained in some \( C \)-component.
Proof. Every vertex of an antihole belongs to a $C_4$ along with every other vertex of the antihole. □

As a result of Properties 5.4 and 5.5, we have the following theorem.

Theorem 5.6. The decomposition into $C$-components is hole and antihole preserving.

The partition induced by $\sim$ on the vertices of $G$ very naturally defines a quotient graph, which we will denote by $G^0$.

Definition 5.7. Let $G = (V, E)$ be an arbitrary graph. Let us define the quotient graph $G^0 = (V', E')$ of $G$, where $V'$ is the set of $C$-components of $G$ and there is an edge between two $C$-components $X_i$ and $X_j$ if there is an edge in $G$ with one endpoint in $X_i$ and the other in $X_j$.

Theorem 5.8. Let $G = (V, E)$ be an arbitrary graph. The quotient graph $G^0$ of $G$ is chordal.

Proof. Suppose graph $G^0$ is not chordal.

There must be a chordless cycle $C^0 = (X_1, \ldots, X_k, X_1)$ in $G^0$. We will construct a corresponding chordless cycle in $G$. Let us consider three consecutive components $X_i$, $X_{i+1}$, and $X_{i+2}$ of $C^0$, and let $x$ be a vertex in $X_{i+1}$; $x$ can see a vertex $x'$ of another component of $C^0$ only if $x'$ is either in $X_i$, or in $X_{i+2}$, else edge $xx'$ corresponds to a chord of $C^0$.

In each component $X_i$ of $C^0$, let us choose some vertex $y_i$ which sees a vertex $x_{i+1}$ of $X_{i+1}$. Thus our construction chooses in each component $X_j$ two vertices, $x_i$ and $y_i$; $x_i$ is seen by $y_{i-1}$. Let $P_i$ be a chordless path in $X_i$ which connects $x_i$ with $y_i$. Let us concatenate all of these paths: we obtain a cycle from which we can extract a chordless cycle $C$ of $G$, which has vertices in different $C$-components, thus contradicting Property 5.4. □

Let us now discuss the case where a vertex does not belong to any chordless cycle. Recall that by Definition 2.13, a vertex is LB-simplicial if all its substars are cliques; by Property 2.12, a vertex is LB-simplicial if and only if it belongs to no chordless cycle. We will express this by the following property.

Property 5.9. Let $X$ be a $C$-component of a graph. The following propositions are equivalent:

1. $X$ contains no chordless cycle.
2. $|X| = 1$.
3. $X$ is an LB-simplicial vertex of the graph.

We will call a $C$-component trivial when it contains no chordless cycle.
Computing the $C$-components of a graph $G$ can be done in $O(m^2)$ time using Definition 5.1. For each edge $xy$ of the graph, one can determine whether it is part of a chordless cycle by removing the edge $xy$ as well as the common neighbors of $x$ and $y$; if in the resulting graph there is a path from $x$ to $y$, then in the original graph $xy$ belongs to a chordless cycle. This test requires $O(m)$ time for each edge, and thus all edges can be tested in $O(m^2)$ time. The $C$-components are then computed as being the connected components of the graph, obtained from $G$ by removing all edges that do not belong to a chordless cycle.

6. Cycle-bicolorable graphs. In this section, we present a new class of perfect graphs which generalizes chordal probe graphs in the case where no partition is given in advance. We exploit the property that in any valid partition, probes and non-probes must alternate on every chordless cycle.

In [11], the following lemma is shown for chordal probe graphs.

**Lemma 6.1** (see [11]). If a graph $G = (P + N, E)$ is chordal probe with respect to the partition \{P, N\} of its vertex set, then probes and non-probes alternate on every chordless cycle of $G$.

We will use this property to introduce a new graph class.

**Definition 6.2.** We will say that a graph $G = (V, E)$ is *cycle-bicolorable* if and only if each vertex can be labeled with one of two colors in such a fashion that the colors alternate in every chordless cycle.

Note that on a $C$-path in a cycle-bicolorable graph, the colors must alternate.

6.1. Recognition of cycle-bicolorable graphs. The following proposition will allow us to characterize cycle-bicolorable graphs by considering each $C$-component separately.

**Proposition 6.3.** A graph is cycle-bicolorable if and only if each of its $C$-components is cycle-bicolorable.

**Proof.** By Property 5.4, every chordless cycle of $G$ is entirely contained in a unique $C$-component of $G$. Thus, coloring the chordless cycles inside each $C$-component is equivalent to coloring all chordless cycles.

**Lemma 6.4.** Each cycle-bicolorable $C$-component has exactly two opposite colorings.

**Proof.** Let us consider a $C$-component $X_i$, and let $\kappa_1$ be a bicoloring of $X_i$. By exchanging the colors of every vertex, another bicoloring $\kappa_2$ is obtained. Suppose there is a third possible coloring $\kappa_3$. Let $x$ be a vertex whose color is different in $\kappa_1$ and $\kappa_3$. We claim that every other vertex in $X_i$ has a different coloring in $\kappa_1$ and in $\kappa_3$. Suppose by contradiction that some vertex $y$ of $X_i$ has the same color in $\kappa_1$ as in $\kappa_3$. There is a $C$-path connecting $x$ and $y$; since the colors in $\kappa_3$ must alternate on this path, the color of $x$ uniquely determines the color of $y$, a contradiction. Thus, the color of every vertex of $X_i$ is different in $\kappa_1$ as in $\kappa_3$, so $\kappa_3$ is the same as $\kappa_2$.

Lemma 6.4 justifies the following definition.

**Definition 6.5.** Let $X$ be a cycle-bicolorable $C$-component of a graph $G$. The bicoloring of $G_X$ induce a unique partition of the vertices into $V_1 + V_2$, which we will call the color bipartition of $X$.

To recognize cycle-bicolorable graphs, we determine the $C$-components as described in section 5, then check that each $C$-component is bicolorable. The correctness follows from Proposition 6.3. Figure 5 gives an easy algorithm to recognize cycle-bicolorable graphs.
Algorithm cycle-bicolorable recognition.

input: A graph $G = (V,E)$, the set $X$ of $C$-components of $G$.
output: “Failure” if $G$ is not bicolorable, otherwise a black/white coloring of the vertices of $G$ such that the colors alternate on every chordless cycle.

//At the beginning, all vertices are uncolored and $Q$ is an empty queue;
while there remains some uncolored vertex do
  //Q is empty.
  Choose a not yet colored $C$-component $\{X_i\}$ of $X$;
  Choose a vertex $x$ of $\{X_i\}$, color it black and insert it into $Q$;
  while $Q$ is non-empty do
    Remove a vertex $y$ from $Q$;
    foreach neighbor $v$ of $y$ in $\{X_i\}$ do
      if $v$ has the same color as $y$'s then
        return (failure);
      if $v$ is uncolored then
        Color $v$ with the color different from $y$'s, and insert $v$ into $Q$;
  return (success);

Complexity. The complexity of recognizing cycle-bicolorable graphs is the same as that of computing the $C$-components of a graph, namely $O(m^2)$.

6.2. Some properties of cycle-bicolorable graphs.

THEOREM 6.6. The class of cycle-bicolorable graphs is perfect.

In order to prove Theorem 6.6, we will need the following lemmas.

LEMMA 6.7. A cycle-bicolorable graph has no odd hole.

Proof. Clearly, an odd chordless cycle cannot be labeled with two colors in a fashion that the colors alternate. □

LEMMA 6.8. A cycle-bicolorable graph has no antihole.

Proof. Let $G = (V,E)$ be a cycle-bicolorable graph. By Lemma 6.7, $G$ has no induced $C_5$, since $C_5$ is isomorphic to $C_5$. Suppose there exists $k \geq 6$, such that $C_k$ is a chordless cycle of $G$, with $C_k = (x_1, \ldots, x_k, x_1)$. Observe that $C' = (x_2, x_4, x_1, x_5, x_2)$ is a cycle of length 4 in $G$. In any bicoloring of $V$ into black and white, $x_1$ and $x_2$ have the same color, w.l.o.g., black. Observe that $x_1$ sees all the vertices in $\overline{C_k}$, except for $x_2$ and $x_k$. Therefore, all the vertices in $\overline{C_k}$, except possibly $x_2$ and $x_k$, are white. But $C'' = (x_3, x_5, x_2, x_6, x_3)$ is also a chordless cycle of length 4 in $G$. In any bicoloring of $V$, either $x_2$ and $x_3$ are black or $x_5$ and $x_6$ are black, a contradiction. □

Theorem 6.6 follows directly from the Strong Perfect Graph Theorem and from Lemmas 6.7 and 6.8.

Remark 6.9. There are graphs with no odd antiholes and no odd holes which are not cycle-bicolorable, as is the case for an even antihole. Figure 6 shows a graph, for which we thank Frédéric Maffray, which has no antiholes and no odd holes, and which is a Meyniel graph and is perfectly orderable, but which is not cycle-bicolorable.

PROPOSITION 6.10. Let $G$ be a cycle-bicolorable graph, where we arbitrarily call $P$ and $N$ the classes induced by a color-bipartition of each $C$-component of $G$. Then
A graph with no antiholes and no odd holes, which is a Meyniel graph and is perfectly orderable, but which is not cycle-bicolorable.

A cycle-bicolorable graph, its C-components, and a corresponding partition into white and black vertices.

$G = (P + N, E)$ is an $N$-$T$ graph.

Note that there are $2^t$ color-bipartitions where $t$ is the number of $C$-components.

Proof of Proposition 6.10. By definition of a cycle-bicolorable graph, in the bicoloring of $G$, there can be no chordless cycle with two consecutive vertices which have the same color; let us arbitrarily call the color classes in each $C$-component of $G$ probes and non-probes: there can be no chordless cycle with two consecutive probes, so by Theorem 3.6, graph $G$ is $N$-$T$ with respect to any partition induced by the bicolorings of its $C$-components.

Note that the converse of Proposition 6.10 does not hold, as $N$-$T$ graphs are not perfect and thus not always cycle-bicolorable, as is the case for the chordless cycle $C_5$.

7. Recognizing non-partitioned chordal probe graphs. From Lemma 6.1, we can easily deduce the following theorem.

Theorem 7.1. Chordal probe graphs are cycle-bicolorable graphs.

The converse fails to hold: the complement of a $P_6$ is cycle-bicolorable but not chordal probe.

In section 6, we saw that a cycle-bicolorable graph can easily be bipartitioned, and we gave an $O(m^2)$ algorithm to do this. We will now apply our results to the recognition of chordal probe graphs.

Lemma 7.2. Let $X_i$ be a $C$-component of a cycle-bicolorable graph. Then $G_{X_i}$ is a chordal probe graph if and only if one of the colors of $X_i$ forms a stable set.

Example 7.3. In Figure 7, the white vertices form a stable set and can be labeled as non-probes; the graph is chordal probe.

Proof of Lemma 7.2. $\Rightarrow$ Let $G_{X_i}$ be a chordal probe graph, and let $P + N$ be a partition of $X_i$ into probes and non-probes where $N$ is a stable set. Since probes and
non-probes alternate on every cycle. \( P + N \) is the unique color bipartition of \( X_i \), by Lemma 6.4. Thus, one of the colors, namely \( N \), is the required stable set.

\( \Leftarrow \) Let \( G_{X_i} \) be cycle-bicolourable such that one of the classes induced by the bicolouring, call it \( N \), is a stable set. By Proposition 6.10, \( G_{X_i} \) is \( N \)-T with respect to this partition, so by definition \( G_{X_i} \) is chordal probe.

In [11, 12], an algorithmic approach was presented to recognize chordal probe graphs in the case where the graph is weakly chordal. We observed in [2] (without proof) that their method (called Procedure “Propagate Constraint Graph”), can also be applied to arbitrary chordal probe graphs, using the following additional lemma. The recognition algorithm that we will present here is a further modification, and we provide a detailed proof.

**Lemma 7.4.** Let \( G \) be a chordal probe graph, let \( X_i \) and \( X_j \) be two bipartite \( C \)-components of the graph. Let \( x \) be a vertex of \( X_i \) which is an endpoint of at least two external edges connecting \( x \) to \( X_j \). Then for any chordal probe partition \( P + N \) of \( V \), \( x \) is a probe.

**Proof.** Let \( Y \) be the set of vertices of \( X_j \) which \( x \) sees. If \( G_Y \) has at least one edge \( e \), then one of the endpoints of \( e \) is a probe and the other a non-probe, since \( X_j \) is bipartite; this forces \( x \) to be a probe. If \( G_Y \) has no edge, then let us choose \( a \), \( b \) and a chordless path \( P \) in \( X_j \) such that \( P \) is shortest possible over all such pairs \( \{a, b\} \); \( P \) together with edges \( ax \) and \( bx \) forms a chordless cycle, which is not fully contained in a \( C \)-component, a contradiction.

**Remark 7.5.** In the case of \( N \)-triangulatable and cycle-bicolourable graphs, Proposition 6.10 showed that it was sufficient to combine any local assignment of \( P + N \) to the cycle-bicolouring of each \( C \)-component to obtain an \( N \)-T graph. This is not the case for chordal probe graphs; we cannot simply apply Lemma 7.2 to each \( C \)-component and combine the results, since globally we must maintain \( N \) as a stable set. For this reason, we must insure that the external edges, which join one \( C \)-component to another, obey the constraint that their endpoints may not both be non-probes.

The considerations described in Remark 7.5 lead us to the algorithm NON-PARTITIONED CHORDAL PROBE GRAPH RECOGNITION presented later which decides whether an arbitrary graph \( G \) is chordal probe and if yes, computes a partition of the vertex set into probes and non-probes. Step 1 checks whether \( G \) is cycle-bicolourable, and if so, produces a bicolouring. Step 2 verifies that each \( C \)-component satisfies Lemma 7.2; if only one color is a stable set, then the labeling into probes and non-probes for that \( C \)-component is fixed by the LABEL-COMPONENT routine; if both colors are stable sets, then that component is a bipartite subgraph and no decision is made (yet) for its labeling. Step 3 applies the condition in Lemma 7.4: for every vertex \( x \) in a unlabeled (hence bipartite) \( C \)-component which sees two vertices in another unlabeled \( C \)-component, the labeling into probes and non-probes for the \( C \)-component containing \( x \) is fixed by the LABEL-COMPONENT routine. Notice that LABEL-COMPONENT has the side effect of building a global queue \( Q \) of external edges which will have to be checked for consistency later in the algorithm.

Step 4 uses the routine PROPAGATE to check the external edges \( uv \) for which \( u \) has been labeled a non-probe, to verify that \( v \) is either a probe or unlabeled; in the latter case, the labeling for the \( C \)-component containing \( v \) is fixed by the LABEL-COMPONENT routine. It terminates when the queue of all such edges is empty. Step 5 simply declares that the graph is now recognized as being chordal probe, although some \( C \)-components may still be unlabeled. Step 6 completes the labeling.
in a greedy manner. Figures 8 and 9 give the subroutines LABEL-COMPONENT and PROPAGATE which are used by the main algorithm.

**Algorithm Non-partitioned chordal probe graph recognition.**

**Input:** A graph $G = (V, E)$.

**Output:** “NO” if $G$ is not chordal probe, otherwise a chordal probe labeling of $V$ with $P$ and $N$.

$Q$ is an empty queue;

**Step 1:** Cycle-bicolorable recognition;

if “failure” is returned then

  return (“NO”);

**Step 2:** foreach C-component $X_i$ do

  if neither black nor white induces a stable set then

    return (“NO”);

  if only one class of the color-bipartition induces a stable set then

    choose a vertex $x$ of this color and label it $N$;

    label-component ($x$, $N$);

  // At this point, the only components left unlabeled are bipartite graphs;

**Step 3:** foreach external edge $x_i x_j$, with $x_i$ in C-component $X_i$ and $x_j$ in C-component $X_j$, $i \neq j$, where $X_i$ and $X_j$ are unlabeled and such that $x_i$ sees at least two vertices in $X_j$ do

  label-component ($x_i$, $P$);

**Step 4:** Propagate;

if “failure” is returned then

  return (“NO”);

  // At this point, any external edge $uv$ with $u$ labeled and $v$ unlabeled
  // will have $u$ labeled as $P$;

**Step 5:** $G$ is chordal probe;

// At this point, some of the C-components may remain unlabeled;

**Step 6:** while there remain some unlabeled C-components do

  Arbitrarily choose an unlabeled vertex $x$;

  label-component ($x$, $P$);

  propagate;

**Complexity.** In the algorithm NON-PARTITIONED CHORDAL PROBE GRAPH RECOGNITION, the bottleneck is Step 1, which requires $O(m^2)$ time. All other steps have lower complexity. An $N$-triangulation can be obtained using the results for the partitioned case.

**Correctness.**

**Theorem 7.6.** The input graph $G$ is chordal probe if and only if algorithm NON-PARTITIONED CHORDAL PROBE GRAPH RECOGNITION does not return “NO.” Moreover, when $G$ is a chordal probe graph, the algorithm produces a valid chordal probe partition $P + N$.

**Proof.** $\Rightarrow$ Suppose $G$ is a chordal probe graph. We show that a “NO” answer gives a contradiction.
Algorithm label-component.

input: A vertex $x$ and its label.

output: Labels the vertices which are in the same $C$-component $X_i$ as $x$ and adds to global queue $Q$ all external edges with one endpoint in $X_i$ labeled $N$.

Label all the vertices of $X_i$ with $P$ or $N$ according to the label of $x$;

foreach external edge $uv$ with $u$ in $X_i$ and labeled $N$ do
  Add $(u, v)$ to $Q$;

Fig. 8. Algorithm LABEL-COMPONENT.

Algorithm propagate.

output: Returns “failure” if there is a conflict, otherwise labels the vertices of some unlabeled $C$-component and may add some to-be-processed edges to the global queue $Q$.

while $Q$ is non-empty do
  $(u, v) \leftarrow$ dequeue($Q$); // $u$ is labeled $N$;
  if $v$ is labeled $N$ then
    return ("failure");
  if $v$ has no label then
    LABEL-COMPONENT($v$, $P$);

Fig. 9. Algorithm PROPAGATE.

If the algorithm returns “NO” in Step 1, then, by Theorem 7.1, $G$ could not be a chordal probe graph. Therefore, Step 1 succeeds and produces a cycle-bicoloring of $G$. If the Algorithm returns “NO” in Step 2, then, by Lemma 7.2, $G$ could not be a chordal probe graph. Therefore, Step 2 succeeds and assigns the probe/non-probe labeling for all non-bipartite $C$-components. Step 3 applies the condition in Lemma 7.4 and always succeeds.

Note that in Steps 2–3, a $C$-component is labeled with a probe/non-probe assignment only when the opposite assignment has been found to be contradictory.

If the Algorithm returns “NO” in Step 4, then the routine PROPAGATE returns “failure” for an external edge whose endpoints were both labeled non-probes, and by Remark 7.5, $G$ could not be a chordal probe graph. Therefore, Step 4 succeeds.

At this point of the algorithm, the following properties hold.

Claim 1. If $X_i$ and $X_j$ are unlabeled $C$-components, then there is at most one edge joining them; we call it the exclusive edge.

Proof of Claim 1. If there were two such external edges $uv$ and $u'v'$ with $u, u' \in X_i$ and $v, v' \in X_j$, by Lemma 7.4, having completed Steps 3–4 we have $u \neq u'$ and $v \neq v'$. From this it follows that the subgraph induced by $uv$ and $u'v'$ and chordless paths in $X_i$ and $X_j$ connecting $u$ with $u'$ and $v$ with $v'$, respectively, contains a chordless
cycle, contradicting Property 5.4.

Claim 2. If \( \{X_1, X_2, \ldots, X_k\} \) are unlabeled \( C \)-components forming a cycle \( C \) in the quotient graph \( G^0 \), i.e., the exclusive edges \( u_i v_{i+1} \) exist joining \( X_i, X_{i+1} \) for all \( i \) (arithmetic mod \( k \)), then \( u_i = v_i \) for all \( i \).

Proof of Claim 2 (by induction on \( k \)). If \( k = 3 \) and one of the equalities fails to hold, then combining shortest paths in \( X_1, X_2, X_3 \) connecting \( u_i \) with \( v_i \), respectively, will yield a chordless cycle in \( G \) of length \( > 4 \), contradicting Property 5.4. If \( k \geq 4 \), then \( C \) has a chord, since the quotient graph \( G^0 \) is chordal (Property 5.8), thus splitting \( C \) into two smaller cycles \( C_1, C_2 \). So by induction, applying Claim 2 to \( C_1, C_2 \), we obtain all \( k \) equalities.

The remainder of the proof of Theorem 7.6 in this direction follows from the next claim.


Proof of Claim 3. Suppose Step 6 fails at an iteration where \( x \) was chosen to be labeled arbitrarily and where failure occurred for external edge \( uv \), where vertex \( u \) is in component \( X_u \) and \( v \) is in \( X_v \), both labeled \( N \). The propagation defines a search tree in the subgraph of the quotient graph induced by the components labeled by that iteration of the propagation. Let \( X_j \) be the smallest common ancestor of \( X_u \) and \( X_v \).

Consider the cycle \( C \) in \( G^0 \) formed by the exclusive edge \( uv \) and the two paths in the search tree from \( X_j \) to \( X_u \) and from \( X_j \) to \( X_v \). Note, however, that the exclusive edges on these paths have the endpoint of the parent labeled \( N \) and the endpoint of the child labeled \( P \), by the routine PROPAGATE. This contradicts Claim 2 and completes the proof of Claim 3.

If the algorithm succeeds, then the probe/non-probe partition \( P + N \) is a cycle-bicoloring (Step 1), so by Proposition 6.10, \( G \) is an \( N \)-T graph with respect to \( P + N \). Furthermore, \( N \) is a stable set since, if \( u, v \in N \) and \( uv \in E \), Step 2 implies \( uv \) is an external edge and PROPAGATE would cause a “failure.” Therefore, by definition, \( G \) is a chordal probe graph, and we have produced a valid chordal probe partition \( P + N \).

8. Conclusion and open questions. Though chordal probe graphs were originally defined as a generalization of interval probe graphs, they may have their own computational biology application as a special case of the chordal graph sandwich problem, which arises in reconstructing phylogenies, tree structures which model genetic mutations, when part of the information is missing [4]. In fact, the polynomiality of \( N \)-T graph recognition which we show in this paper also provides an interesting tractable subcase of the chordal graph sandwich problem [10].

Regarding the structure of chordal probe graphs and \( N \)-T graphs, it appears clearly from the results in this paper that they are similar to chordal graphs in many respects, with similar characterizations. The evident difference is that chordal graphs have no chordless cycles, but we have shown that such cycles can be structured into \( C \)-components, which enables us to handle them efficiently. We believe that \( C \)-components in a general graph may have many interesting properties, one example of which is the chordality of the quotient graph.

We have solved partitioned and non-partitioned chordal probe graph recognition. Our results also solve the problem of non-partitioned interval probe recognition in some subcases, for example, when \( G \) is asteroidal triple free or when the number of \( C \)-components is small. The general non-partitioned interval probe recognition problem has been solved in [5].
Acknowledgment. The authors thank the anonymous referee who pointed out an improvement in the complexity of computing the $C$-components of a graph.

REFERENCES


