Treelike Comparability Graphs: Characterization, Recognition, and Applications

Sabine Cornelsen[1]* and Gabriele Di Stefano[2]

[1] Universität Konstanz, Fachbereich Informatik & Informationswissenschaft, cornelse @ inf.uni-konstanz.de
[2] Università dell'Aquila, Dipartimento di Ingegneria Elettrica, gabriele @ ing.univaq.it

Abstract. An undirected graph is a treelike comparability graph if it admits a transitive orientation such that its transitive reduction is a tree. We show that treelike comparability graphs are distance hereditary. Utilizing this property, we give a linear time recognition algorithm. We then characterize permutation graphs that are treelike. Finally, we consider the Partitioning into Bounded Cliques problem on special subgraphs of treelike permutation graphs.

1 Introduction

An undirected graph is a treelike comparability graph if it admits a transitive orientation such that its transitive reduction is a tree. It is an arborescence, if its transitive reduction is a directed rooted tree. Arborescences were studied by Golumbic [9] and Wolk [15] and characterized as trivially perfect graphs or as graphs that do not contain an induced path of length four nor an induced cycle of length four, respectively. Treelike posets and their linear extension were studied by Atkinson [1].

A graph is completely separable [11] (or distance hereditary) if it can be recursively decomposed into so called splits, such that the remaining components are cliques and stars. The structure of the decomposition is represented in the so called split tree.

In this paper, we first characterize treelike comparability graphs and treelike permutation graphs and give recognition algorithms. We show that a graph is a treelike comparability graph if and only if it is distance hereditary with a special treelike orientation on its split tree. We show how to utilize the split decomposition to recognize treelike comparability graphs in linear time and show that a treelike orientation is unique. Treelike permutation graphs are characterized as paths of arborescence-like graphs and it is shown that the minimum length of such a path can be determined in linear time.

* Work mainly done while the author was visiting the University of L'Aquila, supported by the Human Potential Program of the EU under contract no HPRN-CT-1999-00104 (AMORE Project).

Motivated by train shunting problems [8], we consider the problem PARTITIONING INTO BOUNDED CLIQUES in a second part of this paper, i.e. the problem given $m \in \mathbb{N}$ and a graph $G = (V, E)$, is there a partition of G into cliques of size m? For general graphs, the PARTITIONING INTO BOUNDED CLIQUES-problem is $\mathcal{NP}$-complete for $m \geq 3$ [13] and polynomial time solvable for $m = 2$. It remains $\mathcal{NP}$-complete for comparability graphs and $m \geq 3$ [14], and for permutation graphs and $m \geq 6$ [12]. The complexity of the problem is open for permutation graphs and $m = 3, 4$ or 5. It was shown by Lonc [14] that for fixed m the problem can be solved in linear time on interval graphs. However, it remains $\mathcal{NP}$-complete even for interval graphs if m is part of the input [2]. Bodlaender and Jansen [2] showed that the problem can be solved in $\mathcal{O}(n^{2(m-1)+1})$ time on a graph with n vertices that does not contain an induced path of length four. The problem was considered for many other graph classes. A nice overview can be found, e.g., in [12].

In this paper, we show that the PARTITIONING INTO BOUNDED CLIQUES problem is solvable in linear time for arborescences, even if m is part of the input. We then consider a special matching problem on arborescences and apply its solution to solve the PARTITIONING INTO TRIANGLES-problem in polynomial time on the arborescence-like subgraphs of treelike permutation graphs.

The paper is organized as follows. In Sect. 2, we provide some basic definitions. In Sect. 3, we characterize treelike comparability graphs as special distance hereditary graphs. We utilize this characterization to construct a treelike orientation in linear time. Sect. 4 characterizes treelike permutation graphs. Finally, we consider the PARTITIONING INTO BOUNDED CLIQUES problem on special subgraphs of treelike permutation graphs in Sect. 5.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph. An orientation of E maps each element $\{v, w\}$ of E on exactly one of the ordered pairs (v, w) or (w, v). We refer to the image $\boldsymbol{E}$ of E under a given orientation also as orientation. v is the tail and w is the head of an edge $(v, w) \in \boldsymbol{E}$. Let $v, w \in V$. A $(v - w)$-*path* is a sequence $v, v_1, \ldots, v_{\ell-1}, w$ with $v_1, \ldots, v_{\ell-1} \in V$ distinct vertices and $\{v, v_1\}, \{v_1, v_2\}, \ldots, \{v_{\ell-1}, w\} \in E$. Given an orientation on E, a directed $(v - w)$-path is a path $v, v_1, \ldots, v_{\ell-1}, w$ with $(v, v_1), (v_1, v_2), \ldots, (v_{\ell-1}, w) \in \boldsymbol{E}$. An *(undirected) cycle* is a sequence $v_1, \ldots, v_\ell$ of $\ell > 2$ distinct vertices such that $\{v_1, v_2\}, \ldots, \{v_{\ell-1}, v_\ell\}, \{v_\ell, v_1\} \in E$. A *transitive orientation* is an orientation with the property that there is a directed $(v - w)$-path between two vertices v and w if and only if $(v, w) \in \boldsymbol{E}$. The graph G is a *comparability graph* if there exists a *transitive orientation* on its edges. The *transitive reduction* of a comparability graph G with respect to a fixed transitive orientation $\boldsymbol{E}$ is the spanning subgraph of G that contains exactly the edges of $\boldsymbol{E}$ between two vertices v and w for which there is no directed $(v - w)$-path of length greater than one.

Suppose now that G is a connected comparability graph. A transitive orientation $\boldsymbol{E}$ is called *treelike* if the transitive reduction with respect to $\boldsymbol{E}$ does not contain any undirected cycle. A connected comparability graph is called treelike, if there exists a transitive orientation that is treelike. See Fig. 1 for an example of a comparability graph with two different orientations.

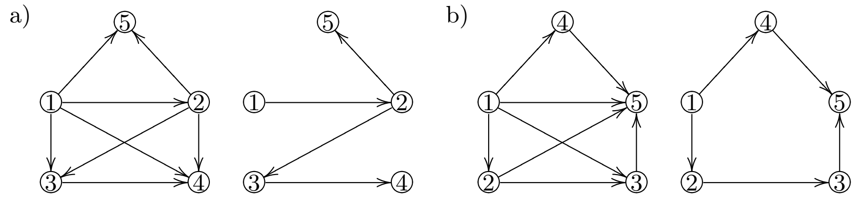

Fig. 1. Transitive reductions with respect to two significantly different transitive orientations of a permutation graph. a) With respect to a treelike orientation b) With respect to an orientation that is not treelike. These two orientations correspond to the two permutations a) $4, 3, 5, 2, 1$ and b) $5, 3, 2, 4, 1$.

Let π be a permutation of $1, \ldots, n$. The *permutation graph* corresponding to π is the graph $G = (V, E)$ with $V = \{1, \ldots, n\}$ and $E = \{\{i, j\};\ i < j \Rightarrow \pi(i) > \pi(j)\}$. It has a transitive orientation $\{(i, j);\ i < j \text{ and } \pi(i) > \pi(j)\}$. We use the representation of a permutation as the graph of the function $i \mapsto \pi(i)$ in the plane, i.e., with the points $(i, \pi(i))$. By definition, there is an edge in the corresponding permutation graph if and only if the slope of the segment between the points $(i, \pi(i))$ and $(j, \pi(j))$ is negative. A graph is a *treelike permutation graph* if it is treelike and a permutation graph.

3 Recognizing Treelike Comparability Graphs

In this section, we show how to construct a treelike orientation of an undirected graph in linear time – if it exists. The algorithm is based on the split decomposition. A *split* of a connected graph $G = (V, E)$ is a partition V_1, V_2 of V into two subsets that have at least two vertices each such that there exist subsets $W_1 \subseteq V_1, W_2 \subseteq V_2$ with the property that the set of edges of G between V_1 and V_2 corresponds to $\{\{w_1, w_2\};\ w_1 \in W_1 \text{ and } w_2 \in W_2\}$. The *split decomposition* of G is defined recursively. Take an arbitrary split $W_1 \subseteq V_1, W_2 \subseteq V_2$. Let the graphs $G_i, i = 1, 2$ be defined as follows. First, consider the subgraph of G induced by V_i. Then add a new vertex w_i – called *special vertex* – with neighborhood W_i. Recursively decompose G_i. The split is memorized in a *special edge* $\{w_1, w_2\}$. The remaining graphs of a split decomposition are called *split components*. The *split tree* associated with a split decomposition is the graph that consists of all split components and all special edges. Let $\{w_1, w_2\}$ be a special edge of a split tree H and let G_1, G_2 be the two split components containing w_1 and w_2. The *re-composition* of G_1 and G_2 is the graph that is obtained from H by deleting w_1 and w_2 and by adding the edges $\{v_1, v_2\}$ for all adjacent vertices $v_1 \neq w_2$ of w_1 and $v_2 \neq w_1$ of w_2. A *minimal split decomposition* of G is a split decomposition of G into three types of components – cliques, stars, and graphs that do not contain a split – such that the number of components is minimized. The minimal split decomposition of a connected graph is unique [5, 6]. G is *completely separable* [11] (or *distance hereditary*) if G can be decomposed such that all split components are cliques or stars.

Before we show that treelike comparability graphs are completely separable, we mention some properties of treelike orientations of the split tree of a completely separable graph. They follow from the facts that a) special edges are not contained in any cycle and b) that the split components of a connected graph are connected and contain at least three vertices.

Remark 1. Let G be a connected graph. Let H be the split tree of G with respect to some split decomposition and assume that H is a treelike comparability graph. The transitive reduction T with respect to a treelike orientation H has the following properties.

1. All special edges are in T.
2. A special vertex is either only the head or only the tail of its incident edges.
3. If two special vertices are adjacent in T, but incident to different special edges, then at least one of them has a degree higher than two in T.

Theorem 1. *Let G be a connected graph and let H be the split tree of G with respect to the minimal split decomposition. Then G is a treelike comparability graph if and only if G is completely separable and there is a treelike orientation with transitive reduction T on H that fulfills the following property.*

(Z) *At most one vertex of each special edge is incident to more than two edges in T.*

Proof. We will show the following properties.

1. Every treelike comparability graph has a split decomposition into components of size three such that the split tree with respect to this decomposition admits a treelike orientation with Property Z.
2. The existence of a treelike orientation on a split tree that fulfills Property Z is maintained under recomposition of two components.

Now, by Property 1, a treelike comparability graph is completely separable. The split components of a minimal split decomposition are obtained by recursively re-composing adjacent cliques or adjacent stars, respectively, in the split tree. Hence, applying recursively Property 2 to the decomposition obtained in 1 yields the only-if direction. Since we obtain the original graph by recursive re-composition, the if-direction follows immediately from Property 2. It remains to show the two properties.

1. Let $G = (V, E)$ be a treelike comparability graph. Let T be the transitive reduction of G with respect to a treelike orientation $\boldsymbol{E}$. We show Property 1 by induction on the number n of vertices of G. There is nothing to show if $n \leq 3$. So let $n > 3$.

If there is an edge of T that is not incident to a leaf of T, let $e = (v_0, v_1)$ be such an edge. Let $v_0 \in V_0, v_1 \in V_1$ be the sets of vertices in the two connected components of $T - e$. Let $W_i, i = 0, 1$ be the set of vertices in V_i that are adjacent to v_{1-i} in G.
If each edge of T is incident to a leave of T, i.e. if T is a star, let r be the central vertex of T. Since T has at least 4 vertices, there are two vertices $v, w \in V$ such that either $(v, r), (w, r)$ are both edges of T or $(r, v), (r, w)$ are both edges of T. Assume $(v, r), (w, r)$. Let $V_0 = \{v, w\}$, $W_0 = V_0$ and $W_1 = \{x \in V;\ (r, x) \text{ in } T\}$.
In either case $\{\{w_0, w_1\};\ w_0 \in W_0 \text{ and } w_1 \in W_1\}$ corresponds to the set of edges of G between V_0 and V_1 and hence V_0, V_1 is a split. Let $G_i, i = 0, 1$ be the subgraphs that result from the decomposition as described above and let $w_i, i = 0, 1$ be the special vertices. Orienting the new edges $(w, w_0), w \in W_0$ and $(w_1, w), w \in W_1$, respectively results in a treelike orientation of G_i with the following new edges in the transitive reduction: (v_0, w_0) and (w_1, v_1) if (v_0, v_1) was chosen as an edge of T non-incident to a leaf and (v, w_0), (w, w_0), and (w_1, r), else. Finally, we orient the special edge (w_1, w_0). Thus, a treelike orientation of the split tree with the required property is maintained in every decomposition step.
2. Let (w_1, w_2) be a special edge. Suppose that w_1 is adjacent to exactly two vertices in T. Let $v \neq w_2$ be the adjacent vertex of w_1 in its component. For each adjacent vertex $w \neq w_1$ of w_2 orient the new edges (w, v) This results in a treelike orientation on the re-composition of the two components containing w_1 and w_2, respectively. The only vertex whose degree might increase is v. But v was incident to a special vertex with degree two. So if v is a special vertex then it already had a degree higher then two. Hence, Prop. Z is maintained. □

The proof of Theorem 1 showed especially that there is the following correspondence between a treelike orientation of a treelike comparability graph and a treelike orientation of its split tree.

Remark 2. Let $\boldsymbol{E}$ be a treelike orientation of a graph G and let H be the split tree of G with respect to the minimal split decomposition. Then there is a treelike orientation $\boldsymbol{E}_H$ of H such that $(v, w) \in \boldsymbol{E}$ if and only if there is an undirected $(v - w)$-path $v = v_0, v_1, \ldots, v_\ell = w$ in H with

- $(v_i, v_{i+1}) \in \boldsymbol{E}_H$ if $\{v_i, v_{i+1}\}$ is not a special edge and
- $(v_{i+1}, v_i) \in \boldsymbol{E}_H$ if $\{v_i, v_{i+1}\}$ is a special edge.

Theorem 2. *It can be tested in linear time whether a graph is a treelike comparability graph. Moreover, let G be a connected treelike comparability graph.*

1. *The treelike orientation of G is unique up to isomorphism and reversing the whole orientation.*
2. *The treelike orientation of G as well as its transitive reduction can be found in linear time.*

Proof. Let G be a connected graph. The following algorithm applied to G outputs the transitive reduction with respect to a treelike orientation of G if G is treelike.

1. Let Q be a queue.
2. Compute the split tree with respect to the minimal split decomposition.
3. If G is not completely separable, G is not treelike. Break.
4. Choose some special edge $\{w_1, w_2\}$ and orient it arbitrarily.
5. Append w_1 and w_2 to Q.
6. While Q is not empty
   (a) Remove the first element w from Q. Suppose (w, w') is a special edge.
   (b) Let H be the split component containing w.
   (c) Orient each edge e of H that is incident to w such that w is the tail of e.
   (d) If H is a star.
      i. If both w and w' are the center of a star, G is not treelike. Break.
      ii. Orient remaining edges such that the center of H is only the head or only the tail of all its incident edges.
   (e) If H is a clique.
      i. If H contains more than two special vertices, G is not treelike. Break.
      ii. Choose an arbitrary ordering $w = v_1, \ldots, v_\ell$ of the vertices of H such that $v_2, \ldots, v_{\ell-1}$ are not special.
      iii. Orient edges (v_i, v_{i+1}) and eliminate remaining edges of H.
   (f) For all special vertices $w_1 \neq w$ of H, let $e = \{w_1, w_2\}$ be a special edge.
      i. If w_1 is the tail of an edge in H orient (w_1, w_2), else (w_2, w_1).
      ii. Append w_2 to Q.
7. Recompose G maintaining only non-transitive edges.

If the algorithm breaks then G is not completely separable (Step 3), or there cannot be a treelike orientation on the split tree that fulfills Property Z in Theorem 1 (Step 6(d)i) or Property 2 in Remark 1 (Step 6(e)i). In either case, G is not treelike.

In Steps 4-6, the algorithm constructs a treelike orientation of the split tree in a breadth first search. By Property 2+3 of Remark 1, there are only two steps in which there is a free choice for the orientation of an edge (Step 4 and Step 6(e)ii). The latter corresponds to choosing the orientation among edges between isomorphic vertices. Hence, a treelike orientation of the split graph of a completely separable graph is unique up to isomorphism and reversing the whole orientation. Thus, if the split tree has a treelike orientation that fulfills Prop. Z of Theorem 1 then the algorithm finds it. This implies the correctness of the algorithm. Uniqueness of the treelike orientation of G follows by Remark 2. □

4 Treelike Permutation Graphs and Arborescences

In this section, we will characterize treelike permutation graphs as paths of double-arborescences. On orientation $\boldsymbol{E}$ of a graph $G = (V, E)$ is an *arborescence-orientation* if the transitive reduction is a rooted tree, i.e., if $\boldsymbol{E}$ is treelike and there is a vertex $r \in V$ such that

$$V = \{r\} \cup \{v \in V;\ (v, r) \in \boldsymbol{E}\} \quad \text{or} \quad V = \{r\} \cup \{v \in V;\ (r, v) \in \boldsymbol{E}\}.$$

$\boldsymbol{E}$ is a *double-arborescence-orientation* if $\boldsymbol{E}$ is treelike and there is a vertex $r \in V$ such that

$$V = \{r\} \cup \{v \in V;\ (v, r) \in \boldsymbol{E}\} \cup \{v \in V;\ (r, v) \in \boldsymbol{E}\}$$

The treelike orientation in Fig. 1a is in fact an arborescence-orientation. We refer to the special vertex r as the root of an arborescence- or a double-arborescence-orientation, respectively. A connected comparability graph is called an arborescence, or a double-arborescence, if there exists an arborescence-, or a double-arborescence-orientation, respectively.

A graph G is a *path of ℓ double-arborescences* if it has a treelike transitive orientation $\boldsymbol{E}$ such that there exists a (not necessarily directed) path P of length $\ell - 1$ in the transitive reduction T that fulfills the following property. Let $V_1, \ldots, V_\ell$ be the vertex sets of the connected components of the graph that results from T by deleting the edges of P. Let $v_i \in V_i, i = 1, \ldots, \ell$ be the vertex in P. Let $G_i, i = 1, \ldots, \ell$ be the subgraphs of G that are induced by V_i. Then $\boldsymbol{E}$ induces a double-arborescence-orientation on $G_i, i = 1, \ldots, \ell$ with root v_i.

To characterize treelike permutation graphs, we apply some results about AT-free graphs. A graph is *AT-free* if it doesn't contain an *asteroidal triple*, i.e. three independent vertices with the property that for every pair of them there is a path connecting the two vertices that does not contain the neighborhood of the remaining vertex. Two vertices u and v are a *dominating pair* of a graph if each vertex of the graph is adjacent to each $(u - v)$-path.

Theorem 3. *Let G be a treelike comparability graph. Then the following are equivalent:*

1. *G is a permutation graph.*
2. *G is AT-free.*
3. *G has a dominating pair.*
4. *G is a path of double-arborescences.*

Proof. $1 \Rightarrow 2$: The complement of a permutation graph is a comparability graph. Hence, a permutation graph is AT-free [10].

$2 \Rightarrow 3$: Every AT-free graph has a dominating pair [4].

$3 \Rightarrow 4$: Let T be the transitive reduction of G with respect to a treelike transitive orientation $\boldsymbol{E}$ of G. Let v, w be a dominating pair of G. Then the unique $(v - w)$-path P in T is a dominating path of G. Hence, for each vertex u of G there has to be a directed path in T from u to a vertex of P. Hence, $\boldsymbol{E}$ induces a double-arborescence orientation on the subgraphs of G that are induced by the connected components of the graph that results from T by deleting the edges of P.

$4 \Rightarrow 1$: Let $\boldsymbol{E}$ be a treelike orientation of G and let T be the transitive reduction of G. Let $P = v_1, \ldots, v_\ell$ be a path of T such that $\boldsymbol{E}$ induces a double-arborescence orientation on the subgraphs of G that are induced by the connected components of the graph that results from T by deleting the edges of P. Let $A_V = \{v \in V;\ (v, v_1) \in \boldsymbol{E}\}$, $B_V = \{v \in V;\ (v_1, v)\}$, and

$C_V = V \setminus (A_V \cup B_V \cup \{v_1\})$. It can be shown by induction on ℓ that G is a permutation graph of a permutation π and that the graph $i \mapsto \pi(i)$ has the following shape:

A_V		C_V
	v_1	
		B_V

□

The previous theorem implies especially that a graph is a treelike permutation graph if and only if it is a path of double-arborescences. In the next theorem, we discuss how to find such a path of minimum length. We will use that a dominating pair of an AT-free graph can be found in linear time [3].

Theorem 4. *Let G be a treelike permutation graph. The minimum ℓ for which G is a path of ℓ double-arborescences can be determined in linear time.*

Proof. Let G be a treelike permutation graph. Let T be the transitive reduction of the unique treelike orientation of G. Let v, w be a dominating pair of G. Find the unique $(v - w)$-path P in T. Recursively remove the first vertex v_1 from P if v_1 is the head of the first edge e of P and the tail of all other edges of T that are incident to v_1 or vice versa. Under the same condition, remove recursively the last vertex of P. Let $\ell - 1$ be the length of the remaining path. Then ℓ is minimum such that G is a path of ℓ double-arborescences. □

5 Partitioning into Bounded Cliques

Let $G = (V, E)$ be a graph. An *m-clique* is a subset $C \subseteq V$ of m vertices, such that $\{v, w\} \in E$ for each pair of vertices $v, w \in C$. A sequence $C_1, \ldots, C_k$ of k cliques is a partition of V into k cliques if $V = C_1 \cup \ldots \cup C_k$ and $C_i \cap C_j = \emptyset, 1 \leq i < j \leq k$. A *triangle* is a 3-clique. We consider the following problem.

Partitioning into m-Cliques: Given a graph $G = (V, E)$, is there a partition of G into m-cliques?

We say that a graph $G' = (V', E')$ results from a graph $G = (V, E)$ by *adding a k-clique* if there are distinct vertices $v_1, \ldots, v_k \notin V$ such that

$$V' = V \cup \{v_1, \ldots, v_k\} \quad \text{and}$$
$$E' = E \cup \{\{v, v_i\};\ v \in V, i = 1, \ldots, k\} \cup \{\{v_i, v_j\};\ 1 \leq i < j \leq k\}.$$

Note that for graph-classes that are closed under adding cliques the Partitioning into m-Cliques problem is equivalent to the following problem: Given a graph $G = (V, E)$ and a number $k \in \mathbb{N}$, is there a partition of G into k cliques of maximum size m? Examples for such graph classes are comparability graphs, permutation graphs, arborescences, and double-arborescences.

In this section, we will show that the Partitioning into m-Cliques problem can be solved in linear time on arborescences – even if m is part of the input. We then show how to compute the maximum number of 2-cliques in an arborescence after deleting some triangles. Finally, we demonstrate how to use these numbers to solve the PARTITIONING INTO TRIANGLES problem on double-arborescences.

5.1 Partitioning Arborescences into Bounded Cliques

Since arborescences are permutation graphs and do not contain an induced cycle of length four, they are especially interval graphs. Recall that the PARTITIONING INTO m-CLIQUES-problem for fixed m can be solved in linear time on interval graphs [14], but that the problem remains $\mathcal{NP}$-complete for interval graphs if m is part of the input [2]. In this section, we give an algorithm that solves the problem partitioning arborescences into bounded cliques in linear time – even if m is part of the input.

Theorem 5. *The problem* PARTITIONING INTO m-CLIQUES *can be solved in linear time on arborescences even if* m *is part of the input.*

Proof. Let $G = (V, E)$ be an arborescence. Let T be the transitive reduction with respect to an arborescence-orientation. By Theorem 2, T can be constructed in linear time. Proceeding from the leaves to the root r of T, we assign a label MISS to each vertex v. Let v be the next vertex of T that is considered. If v is a leave or the only vertex of G, we set $\text{MISS}(v) = m - 1$. Else let $v_1, \ldots, v_k$ be the children of v. If $\sum_{i=1}^{k} \text{MISS}(v_i) = 0$, set $\text{MISS}(v) = m - 1$, else set $\text{MISS}(v) = -1 + \sum_{i=1}^{k} \text{MISS}(v_i)$.

By induction on the number n of vertices of G it follows that $\text{MISS}(r) = k$ if and only if k is the smallest non-negative integer such that adding a k-clique to G results in a graph that has a partition into m-cliques. Hence, G has a partition into m-cliques if and only if $\text{MISS}(r) = 0$. □

5.2 Important Triangles for Maximum Matchings in Arborescences

Throughout this section let $G = (V, E)$ be an arborescence, let T be the transitive reduction of G with respect to an arborescence-orientation, let r be the only sink and let $t_{\max}$ be the maximum number of disjoint triangles of G. A *matching* of a graph $G = (V, E)$ is a subset $M \subseteq E$ of the edge set such that each vertex of G is adjacent to at most one edge in M. Let $\mathcal{V}$ be a set of subsets of V. We denote by $G - \mathcal{V}$ the graph that results from G by deleting all vertices in all sets of $\mathcal{V}$. Let $\mathcal{T}$ be a set of triangles of G. With $c_{\mathcal{T}}$ we denote the maximum size of a matching in $G - \mathcal{T}$. Let $c^{(i)} = \max c_{\mathcal{T}}$, where $\mathcal{T}$ ranges over all disjoint sets of i triangles of G.

Lemma 1. *A set* $\mathcal{T}$ *of* t_{max} *disjoint triangles of* G *with* $c_{\mathcal{T}} = c^{(t_{max})}$ *can be computed in linear time.*

Proof. Let G be an arborescence. Let T be the transitive reduction with respect to an arborescence-orientation. We proceed again from the leaves to the root r

of T. To each node v we assign a list P of 2-cliques and a list S of singletons that is contained in the subtree rooted at v. If v is a leaf, let $P(v) = \emptyset$ and $S(v) = v$. Else let $v_1, \ldots, v_k$ be the children of v.

1. If there is a $j = 1, \ldots, k$ such that $P(v_j)$ contains a 2-clique $\{x, y\}$, delete one 2-clique $\{x, y\}$ from $P(v_j)$ and add the triangle $\{x, y, v\}$ to $\mathcal{T}$.
2. If all lists $P(v_i), i = 1, \ldots, k$ are empty, but there is some $j = 1, \ldots, k$ such that $S(v_j)$ contains a vertex w, delete w from $S(v_j)$, and add $\{w, v\}$ to $P(v_1)$.
3. If all lists $P(v_i), S(v_i), i = 1, \ldots, k$ are empty, add v to $S(v_1)$.

Set $P(v) = P(v_1), \ldots, P(v_k)$ and $S(v) = S(v_1), \ldots, S(v_k)$. Based on the property that vertices in two subtrees of T rooted at distinct children of r are not adjacent in G, it can be easily verified by induction on the number n of vertices of G that $|\mathcal{T}| = t_{\max}$ and that $c_{\mathcal{T}} = c^{(t)}$. □

Note that by omitting Step 1, the algorithm in the proof of Lemma 1 can be used to create a maximum matching $P(r)$. In the remainder of this section, let $\mathcal{T}$ be the set of triangles computed in the proof of Lemma 1.

Theorem 6.

1. *The curve* $i \mapsto c^{(i)}$ *has the* stair shape*, i.e. there are* $1 \leq t_{flat} \leq t_{stair} \leq t_{max}$ *such that* $t_{stair} - t_{flat}$ *is even and*
   (a) $c^{(i-1)} - c^{(i)} = 1, i = 1, \ldots, t_{flat}$
   (b) $c^{(t_{flat}+2i-2)} - c^{(t_{flat}+2i-1)} = 2, c^{(t_{flat}+2i-1)} - c^{(t_{flat}+2i)} = 1, i = 1, \ldots, \frac{t_{stair}-t_{flat}}{2}$
   (c) $c^{(i-1)} - c^{(i)} = 2, i = t_{stair} + 1, \ldots, t_{max}$.
2. *The triangles* $t_1, \ldots, t_{t_{max}}$ *in* $\mathcal{T}$ *can be ordered such that* $c^i = c_{\{t_1,\ldots,t_i\}}, i = 1, \ldots, t_{max}$.
3. *The sequence* $c^{(i)}, i = 1, \ldots, t_{max}$ *can be computed in* $\mathcal{O}(n^4)$ *time.*

For space reasons, we only give the algorithm for Theorem 6.3. The proof of Theorem 6 is basically an induction on the number of steps in the algorithm, but quiet complex. Let t be a triangle of G. Let v be the sink of t. The *subtree of* T *rooted at* t is the subtree T_v of T rooted at v. The *level* of t is the maximum number of disjoint triangles of G that are contained in T_v.

Let $i = 0$. While $i < t_{\max}$ do one of the following cases.

1. If there is a triangle $t \in \mathcal{T} \setminus \{t_1, \ldots t_i\}$ such that $c_{\{t_1,\ldots,t_i\}} - c_{\{t_1,\ldots,t_i,t\}} = 1$, choose t_{i+1} among these triangles on a lowest level. Set $i = i + 1$.
2. Else, if there are two triangles $t, t' \in \mathcal{T} \setminus \{t_1, \ldots t_i\}$ such that $c_{\{t_1,\ldots,t_i\}} - c_{\{t_1,\ldots,t_i,t,t'\}} = 3$, choose t_{i+1}, t_{i+2} among these pairs of triangles. Set $i = i+2$.
3. Else, choose a triangle $t_{i+1} \in \mathcal{T} \setminus \{t_1, \ldots t_i\}$ on a lowest level. Set $i = i + 1$.

5.3 Partitioning into Triangles of Double-Arborescences

Note that double-arborescences are P_4-free (also called cographs), i.e. they do not contain an induced path of length four. Recall that the Partitioning into Triangles problem can be solved in $\mathcal{O}(n^5)$ on a P_4-free graph with n vertices [2]. In this section we show how to apply the results of Section 5.2 to solve the problem in $\mathcal{O}(n^4)$ time on double-arborescences.

Theorem 7. *It can be tested in $\mathcal{O}(n^4)$ time whether a double-arborescence with n vertices has a partition into triangles.*

Proof. Let $G = (V, E)$ be a double-arborescence and let $\boldsymbol{E}$ be a a double-arborescence orientation of G with root r. Let G_1 be the subgraph of G induced by $\{r\} \cup \{w \in V;\ (r, w) \in \boldsymbol{E}\}$ and G_2 be the subgraph of G induced by $\{w \in V;\ (w, r) \in \boldsymbol{E}\}$. Then the connected components of G_1 and G_2 consist of arborescences. Hence, we can compute the maximum numbers t_1 and t_2 of triangles and the numbers $c_1^{(0)}, \ldots, c_1^{(t_1)}$ and $c_2^{(0)}, \ldots, c_2^{(t_2)}$ of remaining maximum matchings in the two subgraphs, respectively. Further let $n_{1,2}$ be the number of vertices in $G_{1,2}$. Now, note that a triangle in G is either a triangle in G_1 or G_2 or it consists of an edge in G_1 and a single vertex in G_2 or vice versa. Hence, there is a partition into triangles of G if and only if there exists a pair $(i, j), 1 \leq i \leq t_1, 1 \leq j \leq t_2$ and $\alpha_1, \alpha_2 \in \mathbb{N}$ with the property that $\alpha_1 \leq c_1^{(i)}$, $\alpha_2 \leq c_2^{(j)}$, and

$$2\alpha_1 + \alpha_2 = n_1 - 3i \text{ and } 2\alpha_2 + \alpha_1 = n_2 - 3j \tag{1}$$

These equations describe the case that there are i triangles in G_1, j triangles in G_2, that α_1 of the maximum $c_1^{(i)}$ remaining 2-cliques of G_1 build a triangle with different singletons in G_2, and that the remaining vertices of G_1 build triangles with α_2 2-cliques of G_2. Resolving the equations for α_1 and α_2 results in the condition that n is a multiple of 3 and that

$$c_1^{(i)} \geq \frac{2}{3}n - n_2 - 2i + j \geq 0 \quad \text{and} \quad c_2^{(j)} \geq \frac{2}{3}n - n_1 - 2j + i \geq 0. \tag{2}$$

Clearly, for each pair (i, j), the conditions in (2) can be tested in constant time. Since the number of pairs (i, j) is at most quadratic in the number of vertices of G, the over all running time of the algorithm, once the values of c are computed, is quadratic. □

6 Conclusion

We characterized treelike comparability graphs as a subclass of completely separable graphs. We showed that the treelike orientation of a treelike comparability graph is unique and that it can be constructed in linear time. We characterized treelike permutation graphs as paths of double-arborescences. We showed that the minimum ℓ such that a given treelike permutation graph is a path of ℓ double-arborescences can be determined in linear time. We then considered the Partitioning into m-Cliques problem. We showed that the problem can be solved in linear time on arborescences even if m is part of the input. Based on an algorithm for finding the maximum size of a matching after deleting some triangles of an arborescence, we gave a polynomial time algorithm for solving the Partitioning into Triangles problem on double-arborescences.

References

1. M. D. Atkinson. On computing the number of linear extensions of a tree. *Order*, 7:23–25, 1990.
2. H. L. Bodlaender and K. Jansen. On the complexity of scheduling incompatible jobs with unit-times. In A. M. Borzyszkowski and S. Sokolowski, editors, *Proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science (MFCS '93)*, volume 711 of *Lecture Notes in Computer Science*, pages 291–300. Springer, 1993.
3. D. G. Corneil, S. Olariu, and L. Stewart. A linear time algorithm to compute a dominating path in an AT-free graph. *Information Processing Letters*, 54(5):253–257, 1995.
4. D. G. Corneil, S. Olariu, and L. Stewart. Astroidal triple-free graphs. *SIAM Journal on Discrete Mathematics*, 10(3):399–430, 1997.
5. W. H. Cunningham and J. Edmonds. A combinatorial decomposition theory. *Canadian Journal of Mathematics*, 32:734–765, 1980.
6. W. H. Cunningham and J. Edmonds. Decomposition of directed graphs. *SIAM Journal on Algebraic and Discrete Methods*, 3:214–228, 1982.
7. E. Dahlhaus. Efficient parallel and linear time sequential split decomposition. In P. S. Thiagarajan, editor, *Proceedings of the 14th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS '94)*, volume 880 of *Lecture Notes in Computer Science*, pages 171–180. Springer, 1994.
8. G. Di Stefano and M. L. Koci. A graph theoretical approach to the shunting problem. In B. Gerards, editor, *Proceedings of the Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS 2003)*, volume 92 of *Electronic Notes in Theoretical Computer Science*, 2004.
9. M. C. Golumbic. Trivially perfect graphs. *Discrete Mathematics*, 24:105–107, 1978.
10. M. C. Golumbic, C. L. Monma, and W. T. Trotter, Jr. Tolerance graphs. *Discrete Applied Mathematics*, 9:157–170, 1984.
11. P. L. Hammer and F. Maffray. Completely seperable graphs. *Discrete Applied Mathematics*, 27:85–99, 1990.
12. K. Jansen. The mutual exclusion scheduling problem for permutation and comparability graphs. In M. Morvan, C. Meinel, and D. Krob, editors, *Proceedings of the 15th Symposium on Theoretical Aspects of Computer Science (STACS '98)*, volume 1373 of *Lecture Notes in Computer Science*, pages 287–297. Springer, 1998.
13. D. G. Kirckpatrick and P. Hell. On the completeness of a generalized matching problem. In *Proceedings of the 10th Annual ACM Symposium on the Theory of Computing (STOC '78)*, pages 240–245. ACM, The Association for Computing Machinery, 1978.
14. Z. Lonc. On complexity of some chain and antichain partition problems. In G. Schmidt and R. Berghammer, editors, *Graph Theoretic Concepts in Computer Science, 17th International Workshop, (WG '91)*, volume 570 of *Lecture Notes in Computer Science*, pages 97–104. Springer, 1992.
15. E. S. Wolk. A note on "The comparability graph of a tree". *Proceedings of the American Mathematical Society*, 16:17–20, 1965.