The vertex leafage of chordal graphs

Steven Chaplick[a], Juraj Stacho[b]

[a]*Department of Physics and Computer Science, Wilfrid Laurier University, 75 University Ave. West, Waterloo, Ontario N2L 3C5, Canada*
[b]*DIMAP and Mathematics Institute, University of Warwick, Coventry CV4 7AL, United Kingdom*

Abstract

Every chordal graph G can be represented as the intersection graph of a collection of subtrees of a host tree, a so-called *tree model* of G. This representation is not necessarily unique. The leafage $\ell(G)$ of a chordal graph G is the minimum number of leaves of the host tree of a tree model of G. The leafage is known to be polynomially computable.

In this contribution, we introduce and study the *vertex leafage*. The vertex leafage $v\ell(G)$ of a chordal graph G is the smallest number k such that there exists a tree model of G in which every subtree has at most k leaves. In particular, the case $v\ell(G) \leq 2$ coincides with the class of path graphs (vertex intersection graphs of paths in trees).

We prove for every fixed $k \geq 3$ that deciding whether the vertex leafage of a given chordal graph is at most k is NP-complete. In particular, we show that the problem is NP-complete on split graphs with vertex leafage of at most $k+1$. We further prove that it is NP-hard to find for a given split graph G (with vertex leafage at most three) a tree model with minimum total number leaves in all subtrees, or where maximum number of subtrees are paths. On the positive side, for chordal graphs of leafage at most ℓ, we show that the vertex leafage can be calculated in time $n^{O(\ell)}$.

Finally, we prove that every chordal graph G admits a tree model that realizes both the leafage and the vertex leafage of G. Notably, for every path graph G, there exists a path model with $\ell(G)$ leaves in the host tree and we describe an $O(n^3)$ time algorithm to compute such a path model.

Key words: chordal graph, leafage, tree model, clique tree, path graph

1. Introduction

In the following text, a graph is always finite, simple, undirected, and loopless. A graph $G = (V, E)$ has vertex set $V(G)$ and edge set $E(G)$. We write uv for the edge $(u, v) \in E(G)$. We use $N_G(v)$ to denote the neighbourhood of v in G, and write $N_G[v] = N_G(v) \cup \{v\}$. The degree of v in G is denoted by $\deg_G(v) = |N_G(v)|$. Where appropriate, we drop the index G, and write $N(v)$, $N[v]$, and $\deg(v)$, respectively. We use $G[X]$ to denote the subgraph of G induced by $X \subseteq V(G)$, and write $G - X$ for the graph $G[V(G) \setminus X]$. We use $G - v$ for $G - \{v\}$. We say that X is a *clique* of G if $G[X]$ is a complete graph, and X is an *independent set* of G if $G[X]$ has no edges.

A *tree model* of a graph $G = (V, E)$ is a pair $\mathscr{T} = (T, \{T_u\}_{u \in V})$ where T is a tree, called a *host tree*, each T_u is a *subtree* of T, and a pair uv is in E if and only if $V(T_u) \cap V(T_v) \neq \emptyset$. In other words, $\mathscr{T}$ consists of a host tree and a collection of its subtrees whose vertex intersection graph is G.

A graph is *chordal* if it does not contain an induced cycle of length four or more. It is well-known [1, 7, 21] that a graph is chordal if and only if it has a tree model. Any chordal graph admits possibly many different tree models.

For a tree T, let $\mathscr{L}(T)$ denote the set of its *leaves*, i.e., vertices of degree one. If T consists of a single node, we define $\mathscr{L}(T) = \emptyset$. In other words, we consider such a tree to have no leaves.

The *leafage* of a chordal graph G, denoted by $\ell(G)$, is defined as the smallest integer ℓ such that there exists a tree model of G whose host tree has ℓ leaves (see [15]). It is easy to see that $\ell(G) = 0$ if and only if G is a complete graph, and otherwise $\ell(G) \geq 2$. Moreover the case $\ell(G) \leq 2$ corresponds precisely to *interval graphs* (intersection graphs of intervals of the real line) [5]. In this sense, the leafage of a chordal graph G measures how close G is to being an interval graph.

Email addresses: chaplick@cs.toronto.edu (Steven Chaplick), j.stacho@warwick.ac.uk (Juraj Stacho)

In this paper, we introduce and study a similar parameter.

Definition 1. For a chordal graph $G = (V, E)$, the *vertex leafage* of G, denoted by $v\ell(G)$, is the smallest integer k such that there exists a tree model $(T, \{T_u\}_{u \in V})$ of G where $|\mathscr{L}(T_u)| \leq k$ for all $u \in V$.

In other words, the vertex leafage of G seeks a tree model of G where each of the subtrees (corresponding to the vertices of G) has at most k leaves and the value of k is smallest possible. (See Figure 1 for illustration.)

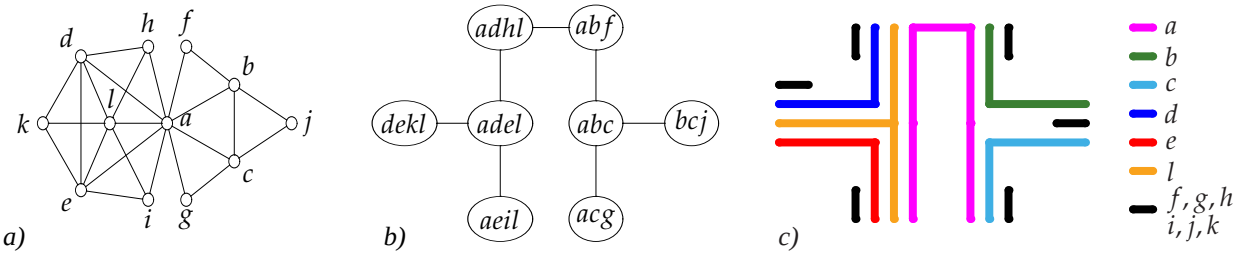

Figure 1: *a)* example graph G with $\ell(G) = 4$ and $v\ell(G) = 3$, *b)* example clique tree T of G, *c)* tree model corresponding to (defined by) T

In the subsequent text, we shall say that a tree model of G *realizes the vertex leafage of* G to indicate that the tree model satisfies the conditions of Definition 1 for smallest possible k. Similarly, we shall say that a tree model of G *realizes the leafage of* G to indicate that the number of leaves in the host tree of the tree model is smallest possible.

As in the case of leafage, the vertex leafage is a natural parameter related to some subclasses of chordal graphs previously studied in the literature. To see this, recall the class of vertex intersection graphs of paths in trees, also known as *path graphs* [8] (see also [2, 14, 16, 18]). Now, observe that for a chordal graph G, we have $v\ell(G) = 0$ if G is a disjoint union of complete graphs, and otherwise $v\ell(G) \geq 2$. Moreover, $v\ell(G) \leq 2$ if and only if G is a path graph. Thus, the vertex leafage of a chordal graph G can be seen as a way to measure how close G is to being a path graph. Another connection comes from [11] where it is observed that in $O(kn)$ time one can find: an optimal colouring, a maximum independent set, a maximum clique, and an optimal clique cover of an n-vertex chordal graph G with vertex leafage k if a representation of G (a tree model realizing vertex leafage) is given.

In [8] it is shown that path graphs can be recognized in polynomial time. Currently, the best known recognition algorithms for path graphs run in $O(nm)$ time [2, 18], where $n = |V(G)|$ and $m = |E(G)|$. In other words, for a chordal graph G, testing whether $v\ell(G) \leq 2$ can be performed in $O(nm)$ time.

Some other restrictions and variations on the standard tree model have also been studied. One such family of these variations is captured by the $[h, s, t]$-graphs (introduced in [13]) defined as follows: $G = (V, E)$ is an $[h, s, t]$-graph if there is a tree model $(T, \{T_u\}_{u \in V})$ of G such that the maximum degree of T is at most h, the maximum degree of each of $\{T_u\}_{u \in V}$ is s, and uv is an edge of G if and only if T_u and T_v have at least t vertices in common. For more information on these graphs see [3, 10].

We summarize the results of our paper in the following theorems.

Theorem 2. *For every $k \geq 3$, it is NP-complete to decide, for a split graph G whose vertex leafage is at most $k+1$, if the vertex leafage of G is at most k.*

Theorem 3. *It is NP-complete to decide, for an integer p and a split graph G whose vertex leafage is at most 3,*

(i) if there exists a tree model of G in which all but p subtrees are paths,
(ii) if there exists a tree model of G where the total number of leaves in all subtrees is at most p.

Theorem 4. *For every $\ell \geq 2$, there exists an $n^{O(\ell)}$ time algorithm that, given an n-vertex chordal graph G with $\ell(G) \leq \ell$, computes the vertex leafage of G and constructs a tree model of G that realizes the vertex leafage of G.*

Theorem 5. *There exists an $O(n^3)$ time algorithm that, given an n-vertex chordal graph $G = (V, E)$ and a tree model $(T, \{T_u\}_{u \in V})$ of G, computes a tree model $(T^*, \{T^*_u\}_{u \in V})$ of G such that*

(i) $|\mathscr{L}(T^*_u)| \leq |\mathscr{L}(T_u)|$ for all $u \in V$,
(ii) $|\mathscr{L}(T^*)| = \ell(G)$.

Corollary 6. *For every chordal graph $G = (V, E)$, there exists a tree model $(T^*, \{T^*_u\}_{u\in V})$ such that*

(i) $|\mathscr{L}(T^*_u)| \leq v\ell(G)$ *for all* $u \in V$.
(ii) $|\mathscr{L}(T^*)| = \ell(G)$,

In other words, such a tree model realizes both the leafage and the vertex leafage of G.

The paper is structured as follows. In §2 we discuss some technical details related to tree models. In §3 we present a proof of Theorem 2 and then discuss how to modify this proof to obtain a proof of Theorem 3. In §4 we prove Theorem 4, and in §5 we present a proof Theorem 5 and Corollary 6. We close the paper in §6 with a summary and a discussion of possible extensions of this work.

2. Minimal Tree Models and Clique Trees

Let $G = (V, E)$ be a chordal graph. We say that two tree models $\mathscr{T} = (T, \{T_u\}_{u\in V})$ and $\mathscr{T}' = (T', \{T'_u\}_{u\in V})$ of G are *isomorphic*, and write $\mathscr{T} \simeq \mathscr{T}'$, if there exists an isomorphism φ between T and T' that induces an isomorphism between T_u and T'_u for all $u \in V$, namely $\varphi\big(V(T_u)\big) = V(T'_u)$.

A tree model $\mathscr{T} = (T, \{T_u\}_{u\in V})$ of G is *minimal* if $|V(T)|$ is smallest possible among all tree models of G. A *clique tree* of G is a tree T whose nodes are the maximal cliques of G such that for all $C, C' \in V(T)$, every C'' on the path between C and C' in T satisfies $C'' \supseteq C \cap C'$. Every clique tree T of G *defines* a tree model $\mathscr{T}_T$ of G, where $\mathscr{T}_T = (T, \{T_u\}_{u\in V})$ and T_u is defined as $T\big[\{C \in V(T) \mid u \in C\}\big]$ for all $u \in V$.

Given an edge XY of a tree T, we denote by $T/_{XY}$ the tree obtained by *contracting* XY in T. Namely, $T/_{XY}$ is the tree constructed by removing X, Y from T, adding a new vertex Z, and connecting to Z the neighbours of X and the neighbours of Y in T. For a tree model $\mathscr{T} = (T, \{T_u\}_{u\in V})$ of G and edge XY of T, by contracting XY in T and all subtrees T_u we mean the tree model with host tree $T/_{XY}$ and subtrees $\{T_u/_{XY} \mid XY \in E(T_u)\} \cup \{T_u \mid XY \notin E(T_u)\}$.

A family of sets $\{X_i\}_{i\in I}$ is said to have the *Helly property* if every pairwise intersecting subfamily of $\{X_i\}_{i\in I}$ has a common point. In other words, every $J \subseteq I$ such that $X_i \cap X_j \neq \emptyset$ for all $i, j \in J$ satisfies $\bigcap_{i\in J} X_i \neq \emptyset$. Note that any collection of subtrees of a tree has the Helly property.

Fact 7. Let $\mathscr{T} = (T, \{T_u\}_{u\in V})$ be a tree model of G. Then the following statements are equivalent.

(i) $\mathscr{T}$ is a minimal tree model of G.
(ii) $\mathscr{T} \simeq \mathscr{T}_T$ for some clique tree T of G.
(iii) For all $XY \in E(T)$, contracting XY in T and all subtrees T_u containing it yields a tree model of $G' \neq G$.
(iv) The mapping ψ defined for $X \in V(T)$ as $\psi(X) = \{u \in V \mid X \in V(T_u)\}$ is a bijection between the vertices of T and the maximal cliques of G.

Proof. (i)⇒(iii) and (ii)⇔(iv) are clear, while (iii)⇒(iv)⇒(i) follow from the Helly property of subtrees. □

Note that Fact 7(iv) states, in other words, that the set of all vertices of G whose subtrees contain X is a maximal clique of G. In particular, for any tree model, the set of such vertices is always a clique of G, but it is not always necessarily a maximal clique. This is only true for minimal tree models.

Moreover, it follows from Fact 7(i)⇔(iii) that every tree model $(T, \{T_u\}_{u\in V})$ of G can be transformed (by contracting some edges of the host tree and the subtrees) into a minimal tree model $(T', \{T'_u\}_{u\in V})$. Notably, as this transformation involves only contracting edges, it follows that this does not increase the number of leaves both in the host tree and the subtrees, namely $|\mathscr{L}(T')| \leq |\mathscr{L}(T)|$ and $|\mathscr{L}(T_u)| \leq |\mathscr{L}(T'_u)|$ for all $u \in V$.

This observation allows us to focus exclusively on minimal tree models. Namely, it shows that if there exists a tree model with minimum number of leaves in the host tree (subtrees), then there also is a minimal tree model with minimum number of leaves in the host tree (subtrees). Consequently, in the remainder of the paper, all tree models are assumed to be minimal tree models unless otherwise specified.

Furthermore, using Fact 7(i)⇔(ii), we shall view minimal tree models of G as tree models defined by clique trees of G. We shall switch between the two viewpoints as needed.

3. Hardness of Vertex Leafage

In this section, we first prove Theorem 2 stating that calculating the vertex leafage of a split graph is NP-complete. To this end, we describe a polynomial-time reduction from NOT-ALL-EQUAL-k-SAT; this problem is well-known to be NP-complete [6] for $k \geq 3$. We then use the same transformation for $k = 2$ to produce a polynomial-time reduction from MAX-CUT, which will prove Theorem 3.

Proof of Theorem 2. The problem is clearly in NP as one can easily compute in polynomial time the number of leaves in subtrees of a given tree model. To prove NP-hardness, we show a reduction from NOT-ALL-EQUAL-k-SAT. By standard arguments [17], we may assume, without loss of generality, that the instances to this problem contain no repeated literals and no negated variables. Thus we can phrase the problem as follows.

NOT-ALL-EQUAL-k-SAT

Instance $\mathcal{I}$: a collection $C_1, C_2, \ldots, C_m$ of k-element subsets of $\{v_1, \ldots, v_n\}$;
Solution to $\mathcal{I}$ (if exists): a set $S \subseteq \{v_1, \ldots, v_n\}$ such that each $j \in \{1, \ldots, m\}$ satisfies $C_j \cap S \neq \emptyset$ and $C_j \setminus S \neq \emptyset$.

In addition, we may assume the following property of any instance $\mathcal{I}$.

($\star$) There are no distinct indices i, i^+ such that $v_{i^+} \in C_j$ whenever $v_i \in C_j$.

Indeed, if there exist $i \neq i^+$ with $v_{i^+} \in C_j$ whenever $v_i \in C_j$, then we replace $\mathcal{I}$ by another instance $\mathcal{I}^+$ constructed from $\mathcal{I}$ by removing v_i and all clauses C_j that contain v_i. If there is a solution to $\mathcal{I}$, then $S \setminus \{v_i\}$ is a solution to $\mathcal{I}^+$. Conversely, if S is a solution to $\mathcal{I}^+$, then either S is a solution to $\mathcal{I}$ if $v_{i^+} \in S$, or $S \cup \{v_i\}$ is a solution to $\mathcal{I}$ if $v_{i^+} \notin S$.

Now, for the reduction, we consider an instance $\mathcal{I}$ satisfying ($\star$) and construct a graph, denoted by $G_\mathcal{I}$, as follows:

(i) the vertex set of $G_\mathcal{I}$ consists of $n + m + 2$ vertices: $V(G_\mathcal{I}) = \{v_1, \ldots, v_n, y_1, \ldots, y_m, z_1, z_2\}$,
(ii) the vertices $\{y_1, \ldots, y_m\}$ form a clique,
(iii) the vertices $\{v_1, \ldots, v_n, z_1, z_2\}$ form an independent set,
(iv) each vertex v_i is adjacent to all vertices y_j such that $v_i \in C_j$,
(v) the vertices z_1, z_2 are adjacent to each vertex of the clique $\{y_1, \ldots, y_m\}$.

We observe that $G_\mathcal{I}$ is a split graph with partition into clique $\{y_1, \ldots, y_m\}$ and independent set $\{v_1, \ldots, v_n, z_1, z_2\}$. (See Figure 2 for an example of this construction.)

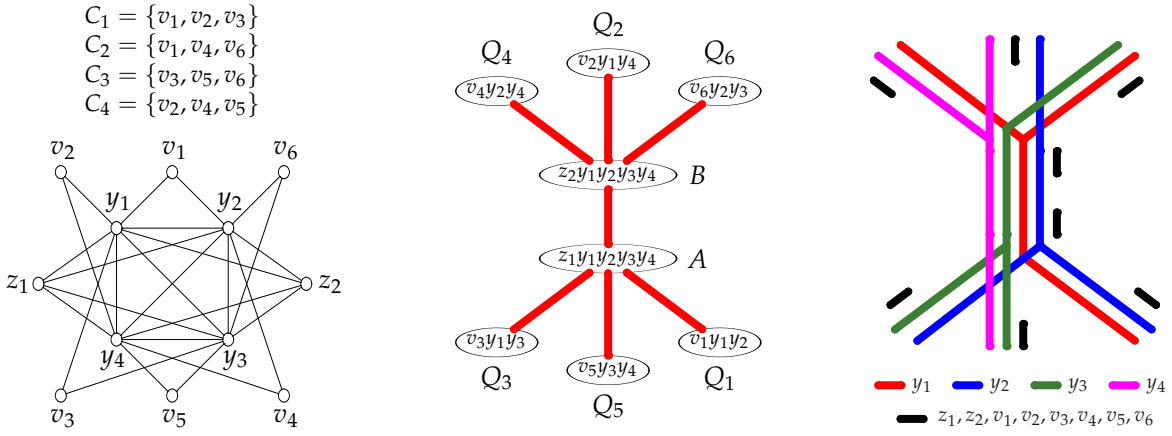

Figure 2: *a)* the graph $G_\mathcal{I}$ for example instance $\mathcal{I}$ with $n = 6$ and $m = 4$, *b)* a clique tree of $G_\mathcal{I}$, *c)* corresponding tree model of $G_\mathcal{I}$.

We prove that the vertex leafage of $G_\mathcal{I}$ is:

(a) at most $k + 1$, and
(b) is at most k if and only if there is a solution to $\mathcal{I}$.

To do this we analyze the cliques of $G_\mathcal{I}$. This is easy, since $G_\mathcal{I}$ is a split graph; all its maximal cliques are formed by taking a vertex of the independent set with its neighbourhood. In particular, the maximal cliques of $G_\mathcal{I}$ are $A = \{z_1, y_1, \ldots, y_m\}$, $B = \{z_2, y_1, \ldots, y_m\}$, and $Q_i = \{v_i\} \cup \{y_j \mid v_i \in C_j\}$ for each $i \in \{1, \ldots, n\}$.

We first prove (a). Recall that $\{A, B, Q_1, \ldots, Q_n\}$ is the set of all maximal cliques of $G_{\mathcal{I}}$, and hence, the vertex set of every clique tree of $G_{\mathcal{I}}$. Each of the vertices z_1, z_2, and v_i, for $i \in \{1, \ldots, n\}$, belongs to exactly one of these cliques, namely A, B, and Q_i, respectively. Also, each y_j, for $j \in \{1, \ldots, m\}$, belongs to exactly $k+2$ cliques, namely A, B, and $\{Q_{i_1}, \ldots, Q_{i_k}\}$ where $C_j = \{v_{i_1}, \ldots, v_{i_k}\}$. So, since $k \geq 1$, every tree spanning these cliques has at most $k+1$ leaves. We thus conclude that in every clique tree of $G_{\mathcal{I}}$, each subtree corresponding to a vertex of $G_{\mathcal{I}}$ has at most $k+1$ leaves. In other words, any clique tree of $G_{\mathcal{I}}$ certifies that $v\ell(G_{\mathcal{I}}) \leq k+1$ which proves (a).

We now prove (b). Let S be a solution to $\mathcal{I}$. Construct a tree T with vertex set $\{A, B, Q_1, \ldots, Q_n\}$ and edge set $\{AB\} \cup \{AQ_i \mid v_i \in S\} \cup \{BQ_i \mid v_i \notin S\}$. Let us verify that T is a clique tree of $G_{\mathcal{I}}$. Its vertex set is the set of all maximal cliques of $G_{\mathcal{I}}$. For distinct $i, i^+ \in \{1, \ldots, n\}$, the path between Q_i and Q_{i^+} contains A or B or both, and no other vertex. Note that $Q_i \cap Q_{i^+} \subseteq \{y_1, \ldots, y_m\} = A \cap B$. This verifies the path between Q_i and Q_{i^+}. Similarly, the path between Q_i and A or B additionally contains only A or B and we have $Q_i \cap A = Q_i \cap B$ which verifies this path. That exhausts all paths in T and thus confirms that T is indeed a clique tree of $G_{\mathcal{I}}$.

Let $\mathscr{T}_T = \left(T, \{T_v\}_{v \in V(G_{\mathcal{I}})}\right)$ be the tree model corresponding to T. We analyze its subtrees. First, we consider the subtree T_{v_i} where $i \in \{1, \ldots, n\}$. As in (a), we observe that the vertex v_i only belongs to one clique of $G_{\mathcal{I}}$, namely Q_i. Thus $|V(T_{v_i})| = 1$ implying $|\mathscr{L}(T_{v_i})| = 0$ by our convention. Similarly, the vertices z_1 and z_2 each belong to only one clique, A and B respectively, and we have $|\mathscr{L}(T_{z_1})| = |\mathscr{L}(T_{z_2})| = 0$. It remains to consider T_{y_j} for $j \in \{1, \ldots, m\}$. The vertex y_j belongs to the cliques A, B, and k distinct cliques $Q_{i_1}, \ldots, Q_{i_k}$ where $C_j = \{v_{i_1}, \ldots, v_{i_k}\}$. The cliques $Q_{i_1}, \ldots, Q_{i_k}$ are leaves of T_{y_j} as they are leaves of T. However, neither A nor B is a leaf of T_{y_j}. Indeed, since S is a solution to $\mathcal{I}$, there are indices $p, r \in \{1, \ldots, k\}$ such that $v_{i_p} \in S$ and $v_{i_r} \notin S$. Hence, by construction, T contains edges AQ_{i_p} and BQ_{i_r}. So, T_{y_j} contains these edges, as well as the edge AB. Thus both A and B have at least two neighbours in T_{y_j} and are therefore not leaves of T_{y_j}. Consequently, $|\mathscr{L}(T_{y_j})| = |\{Q_{i_1}, \ldots, Q_{i_k}\}| = k$ which implies $v\ell(G_{\mathcal{I}}) \leq k$ as certified by the tree model $\mathscr{T}_T$.

Conversely, suppose that $v\ell(G_{\mathcal{I}}) \leq k$. Then there exists a clique tree T of $G_{\mathcal{I}}$ such that the corresponding model $\mathscr{T}_T = \left(T, \{T_v\}_{v \in V(G_{\mathcal{I}})}\right)$ satisfies $|\mathscr{L}(T_v)| \leq k$ for all $v \in V(G_{\mathcal{I}})$. We analyze the structure of T. First, we observe that AB must be an edge of T. If otherwise, the path between A and B in T contains some clique Q_i, $i \in \{1, \ldots, n\}$. As T is a clique tree, we conclude $\{y_1, \ldots, y_m\} = A \cap B \subseteq Q_i = \{v_i\} \cup \{y_j \mid v_i \in C_j\}$. But then v_i belongs to each C_j, $j \in \{1, \ldots, m\}$, and since $n \geq k \geq 2$, there exists $i^+ \in \{1, \ldots, n\}$ different from i, which contradicts ($\star$). Similarly, we show that each Q_i, $i \in \{1, \ldots, n\}$ is a leaf of T. If otherwise, some Q_i has at least two neighbours in T. These cannot be A, B as this would imply a triangle in T, since AB is an edge of T. Thus Q_i is adjacent to Q_{i^+} for some $i^+ \in \{1, \ldots, n\}$. As T is a tree, we have that either Q_{i^+} lies on the path from A to Q_i, or Q_i lies on the path from A to Q_{i^+}. By symmetry, we may assume the former. Thus, since T is a clique tree, we conclude $\{y_j \mid v_i \in C_j\} = A \cap Q_i \subseteq Q_{i^+} = \{v_{i^+}\} \cup \{y_j \mid v_{i^+} \in C_j\}$. So $v_{i^+} \in C_j$ whenever $v_i \in C_j$, contradicting ($\star$).

Now, we are ready to construct a set $S \subseteq \{v_1, \ldots, v_n\}$ as follows: for each $i \in \{1, \ldots, n\}$, we put v_i in S if AQ_i is an edge of T. We show that S is a solution to $\mathcal{I}$. If not, there exists $j \in \{1, \ldots, m\}$ such that either $S \supseteq C_j$ or $S \cap C_j = \emptyset$. We look at the subtree T_{y_j} corresponding to the vertex y_j. Recall that y_j belongs to cliques A, B, and k cliques $Q_{i_1}, \ldots, Q_{i_k}$ where $C_k = \{v_{i_1}, \ldots, v_{i_k}\}$. The cliques $Q_{i_1}, \ldots, Q_{i_k}$ are leaves of T_{y_j} because they are leaves of T (as proved above). If $S \subseteq C_j$, we have, by construction, that A is the unique neighbour of each of the cliques $Q_{i_1}, \ldots, Q_{i_k}$ in T. Consequently, none of the cliques $Q_{i_1}, \ldots, Q_{i_k}$ is adjacent to B in T. This shows that B is only adjacent to A in T_{y_j}, and hence, is a leaf in T_{y_j}. But then $|\mathscr{L}(T_{y_j})| = |\{Q_{i_1}, \ldots, Q_{i_k}, B\}| = k+1$, contradicting our assumption about T. Similarly, if $S \cap C_j = \emptyset$, the cliques $Q_{i_1}, \ldots, Q_{i_k}$ are only adjacent to B and not to A, in which case, A is a leaf of T_{y_j} leading to the same contradiction.

Therefore, S must indeed be a solution to $\mathcal{I}$ and that concludes the proof. □

Proof of Theorem 3. The proof follows the same steps as that of Theorem 2. To keep things simple, we only describe here the key differences. Instead of NOT-ALL-EQUAL-k-SAT, we consider the optimization version of the problem for $k = 2$, which is known as MAX-CUT; this problem is also known to be NP-hard (see [6]).

In MAX-CUT, we are given a graph H, and we seek a subset S of vertices of H such that the number of edges between S and the rest of H is maximized. To be able to reuse our proof of Theorem 2, we cast this problem as follows.

MAX-CUT
Instance $\mathcal{I}$: a collection $C_1, C_2, \ldots, C_m$ of 2-element subsets of $\{v_1, \ldots, v_n\}$, and an integer p
Solution to $\mathcal{I}$: a set $S \subseteq \{v_1, \ldots, v_n\}$ such that $C_j \cap S \neq \emptyset$ and $C_j \setminus S \neq \emptyset$ for at least $m - p$ indices $j \in \{1, \ldots, m\}$

As in the proof of Theorem 2, we may assume that every given instance of this problem satisfies the property ($\star$). (Note that this corresponds to removing vertices of degree zero and one in the corresponding input graph, which does not change hardness of the problem.)

For the reduction, consider an instance $\mathcal{I}$ of the above problem, namely, a collection $C_1, C_2, \ldots, C_m$ of 2-element subsets of $\{v_1, \ldots, v_n\}$ and an integer p. Let $G_{\mathcal{I}}$ denote the graph constructed in the proof of Theorem 2 for the collection $\{C_1, \ldots, C_m\}$. To be more specific, $G_{\mathcal{I}}$ is the graph whose vertex set is $\{v_1, \ldots, v_n, y_1, \ldots, y_m, z_1, z_2\}$ where $\{v_1, \ldots, v_n, z_1, z_2\}$ forms an independent set, $\{y_1, \ldots, y_m\}$ forms a clique, and both z_1 and z_2 are adjacent to all of $y_1, \ldots, y_m$, while v_i is adjacent to y_j if and only if $v_i \in C_j$. Note that $G_{\mathcal{I}}$ is a split graph.

For this graph $G_{\mathcal{I}}$, repeating the arguments presented in the proof of Theorem 2, we conclude that

(a) $v\ell(G_{\mathcal{I}}) \leq 3$, and
(b) there exists a solution to $\mathcal{I}$ if and only if there exists a tree model of $G_{\mathcal{I}}$ in which all but p subtrees are paths.

Moreover, we may assume that the tree model mentioned in (b) is a minimal tree model, and thus (a) implies that the total number of leaves in this model is at most p. This follows from the fact that each of the vertices $v_1, \ldots, v_n, z_1, z_2$ belongs to exactly one maximal clique of $G_{\mathcal{I}}$, and hence, the subtrees corresponding to these vertices have no leaves, since the model is minimal. This proves both parts of Theorem 3 and completes the proof. □

4. Vertex Leafage in Bounded Leafage Graphs

In this section, we discuss calculating vertex leafage in chordal graphs of bounded leafage. Namely, we prove Theorem 4, that is, for a fixed ℓ, we demonstrate how to calculate the vertex leafage of an n-vertex chordal graph G with $\ell(G) \leq \ell$ in polynomial time, namely, in time $n^{O(\ell)}$. We do this by enumerating clique trees of G with respect to high (≥ 3) degree nodes. The enumeration is based on the observation that the number of high-degree nodes in a tree is directly related to the number of leaves. This goes as follows.

For a tree T, let $\mathscr{H}(T)$ denote the set of nodes of T of degree at least 3, and let $\mathscr{E}(T)$ denote the set of edges of T incident to the nodes in $\mathscr{H}(T)$. Further, let n_i denote the number of nodes of degree i in T. Then

$$(\star\star) \quad \begin{aligned} |\mathscr{H}(T)| &= \sum_{i\geq 3} n_i \leq \sum_{i\geq 3}(i-2)n_i = n_1 + \sum_{i\geq 1}(i-2)n_i = n_1 + 2|E(T)| - 2|V(T)| = \ |\mathscr{L}(T)| - 2 \\ |\mathscr{E}(T)| &\leq \sum_{i\geq 3}(i\cdot n_i) \ = \ 2\sum_{i\geq 3} n_i + \sum_{i\geq 3}(i-2)n_i \ = \ 2|\mathscr{H}(T)| + |\mathscr{L}(T)| - 2 \ \leq \ 3|\mathscr{L}(T)| - 6 \end{aligned}$$

For the second-to-last equality in the first line, note that $|V(T)| = \sum_{i\geq 1} n_i$ while $|E(T)| = \frac{1}{2}\sum_{i\geq 1}(i\cdot n_i)$.

We remark that ($\star\star$), in particular, implies that if $|\mathscr{L}(T)|$ is bounded, then so are $|\mathscr{H}(T)|$ and $|\mathscr{E}(T)|$. We shall use this fact later. Moreover we shall use the following property.

Lemma 8. *If T and T^* are two clique trees of G with $\mathscr{E}(T) = \mathscr{E}(T^*)$, then $|\mathscr{L}(T_u)| = |\mathscr{L}(T_u^*)|$ for all $u \in V(G)$ where $T_u = T\big[\{C \in V(T) \mid u \in C\}\big]$ and $T_u^* = T^*\big[\{C \in V(T^*) \mid u \in C\}\big]$.*

Proof. Consider a vertex $u \in V(G)$. Since $\mathscr{E}(T) = \mathscr{E}(T^*)$, we conclude $\mathscr{H}(T) = \mathscr{H}(T^*)$ and each $C \in \mathscr{H}(T) = \mathscr{H}(T^*)$ has the same neighbourhood in both T and T^*, i.e., $N_T(C) = N_{T^*}(C)$. Moreover, if a node has degree at least 3 in T_u, then it also has degree at least 3 in T, since T_u is an subgraph of T. In other words, we have $\mathscr{H}(T_u) \subseteq \mathscr{H}(T)$. In addition, we observe that each $C \in V(T_u)$ satisfies $N_{T_u}(C) = N_T(C) \cap V(T_u)$, since T_u is an induced subgraph of T. By the same token, $N_{T_u^*}(C) = N_{T^*}(C) \cap V(T_u^*)$ for each $C \in V(T_u^*)$. Finally, note that $V(T_u) = V(T_u^*)$, since $V(T) = V(T^*)$. Thus, for each $C \in \mathscr{H}(T_u)$, we can write

$$N_{T_u}(C) = N_T(C) \cap V(T_u) = N_{T^*}(C) \cap V(T_u^*) = N_{T_u^*}(C).$$

This implies $C \in \mathscr{H}(T_u^*)$ and $\deg_{T_u}(C) = \deg_{T_u^*}(C)$ for all $C \in \mathscr{H}(T_u)$. Thus, we calculate by ($\star\star$).

$$|\mathscr{L}(T_u)| = 2 + \sum_{C\in\mathscr{H}(T_u)} \big(\deg_{T_u}(C) - 2\big) \leq 2 + \sum_{C\in\mathscr{H}(T_u^*)} \big(\deg_{T_u^*}(C) - 2\big) = |\mathscr{L}(T_u^*)|$$

To see that the inequality holds, also note that $\deg_{T_u^*}(C) \geq 3$ for each $C \in \mathscr{H}(T_u^*)$, by definition. This proves that $|\mathscr{L}(T_u)| \leq |\mathscr{L}(T_u^*)|$, and a symmetric argument yields $|\mathscr{L}(T_u)| \geq |\mathscr{L}(T_u^*)|$ which completes the proof. □

Now, recall that the vertex set of every clique tree of G is the set of all maximal cliques of G. Notably, all clique trees have the same vertex set. Let $\mathscr{C}(G)$ denote the clique graph of G, i.e., the graph whose nodes are the maximal cliques of G and where two nodes are adjacent if and only if the corresponding maximal cliques intersect. It is well-known [9, 19] that every clique tree of G is a spanning tree of $\mathscr{C}(G)$.

Our algorithm is based on the following lemma.

Lemma 9. *There is an $O(n^3)$ time algorithm that, given an n-vertex chordal graph G and a set $F \subseteq E(\mathscr{C}(G))$, decides if there exists a clique tree T of G with $\mathscr{E}(T) = F$ and constructs such a tree if one exists.*

Proof. We describe an algorithm for the problem as follows.

Algorithm 1:

Input: A chordal graph G and a set $F \subseteq E(\mathscr{C}(G))$.
Output: A clique tree T of G with $\mathscr{E}(T) = F$, or report that no such tree exists.

1 Construct a graph G' as follows:
$V(G') = V(G) \cup \{v_e \mid e \in F\}$
$E(G') = E(G) \cup \{uv_e \mid e \in F,\, e = CC',\, u \in C \cup C'\} \cup \{v_e v_{e'} \mid e, e' \in F,\, e \cap e' \neq \emptyset\}$
2 **if** G' is chordal **then**
3 Construct a clique tree T' of G' with minimum number of leaves.
4 Construct a tree T from T' by renaming each node $C' \in V(T')$ to $C' \cap V(G)$
5 **if** T is a clique tree of G and $\mathscr{E}(T) = F$ **then**
6 **return** T
7 **return** "no such tree exists"

We now prove correctness of the above algorithm. For simplicity, we shall refer to any clique tree T with $\mathscr{E}(T) = F$ as a "solution". First, observe that if the algorithm returns the tree T in Line 6, then this is indeed a solution. This proves that if there is no solution, the algorithm provides the correct answer (in Line 7).

Thus, for the rest of the proof, we may assume that a solution exists. Namely we shall assume there is a clique tree T^* of G satisfying $\mathscr{E}(T^*) = F$. For every maximal clique C of G, define $\varphi(C) = C \cup \{v_e \mid e \in F,\, C \in e\}$.

In the following claim, we discuss the properties of the graph G' constructed in Line 1.

(1) G' is chordal, satisfies $\ell(G') \leq |\mathscr{L}(T^*)|$, and φ is a bijection between the maximal cliques of G and G'.

To prove the claim, we construct a minimal tree model of G' as follows. Let $\mathscr{T}_{T^*} = (T^*, \{T^*_u\}_{u \in V(G)})$ be the minimal tree model of G that is defined by the clique tree T^*, namely $T^*_u = T[\{C \in V(T^*) \mid u \in C\}]$. For each edge $e = CC' \in F$, define $T^*_{v_e} = T^*[\{C, C'\}]$. Finally, let $\mathscr{T}^+ = (T^*, \{T^*_u\}_{u \in V(G)} \cup \{T^*_{v_e}\}_{e \in F})$.

It is easy to verify that $\mathscr{T}^+$ is a tree model of G'. In particular, each subtree in the collection is a connected subgraph of T^*. This follows from the fact that T^* is a clique tree of G and that $F = \mathscr{E}(T^*) \subseteq E(T^*)$. Further, for each edge $e = CC'$ in F, we see that the subtree $T^*_{v_e}$ intersects only subtrees T^*_u where C or C' is in $V(T^*_u)$, i.e., those where $u \in C \cup C'$. Moreover, $T^*_{v_e}$ only intersects subtrees $T^*_{v_{e'}}$ where C or C' is in $V(T^*_{v_{e'}})$, i.e., those where $e \cap e' \neq \emptyset$. This corresponds precisely to the definition of G'.

Thus, we conclude that G' is indeed a chordal graph, and $\ell(G') \leq |\mathscr{L}(T^*)|$ as $\mathscr{T}^+$ is a particular tree model of G' and T^* is its host tree. Morever, we see that $\mathscr{T}^+$ is actually a minimal tree model of G'. Indeed, if there were a tree model of G' with less than $|V(T^*)|$ nodes in its host tree, then by removing subtrees corresponding to the vertices $\{v_e \mid e \in F\}$ we would obtain a tree model of G whose host tree has less than $|V(T^*)|$ nodes. But this would contradict the minimality of $\mathscr{T}_{T^*}$.

This implies, by Fact 7(ii), that there exists a clique tree T^+ of G' that defines $\mathscr{T}^+$, i.e., $\mathscr{T}^+ = \mathscr{T}_{T^+}$. Namely, there is an isomorphism between T^+ and the host tree T^* of $\mathscr{T}^+$ where each node $C \in V(T^*)$ corresponds to the set of all vertices of G' whose subtrees contain C, i.e., the set $\{u \in V(G) \mid C \in V(T_u)\} \cup \{v_e \mid e \in F,\, C \in V(T_{v_e})\}$ which is exactly $\varphi(C)$. In other words, $V(T^+) = \{\varphi(C) \mid C \in V(T^*)\}$, and consequently, φ constitutes an isomorphism between T^* and T^+. As one is a clique tree of G and the other a clique tree of G', we conclude that φ is a bijection between the maximal cliques of G and G'. This proves (1).

The claim (1) shows that the test in Line 2 succeeds. Now, consider the trees T' and T constructed in Line 3 and Line 4, respectively. Note that T' is a clique tree of G' with $|\mathscr{L}(T')| = \ell(G')$.

(2) T is a clique tree of G.

Recall that T is obtained from T' by renaming each node C' of T' to $C' \cap V(G)$. Moreover, by (1), the mapping φ is a bijection between the maximal cliques of G and G'. Namely, for each $C' \in V(T')$, the set $C = \varphi^{-1}(C')$ is a maximal clique of G. Therefore, we can write

$$C' \cap V(G) = \varphi(C) \cap V(G) = \big(C \cup \{v_e \mid e \in F,\, C \in e\}\big) \cap V(G) = C = \varphi^{-1}(C').$$

This proves that the vertex set of T is precisely the set of maximal cliques of G, and φ is an isomorphism between T and T', by the construction of T. To see that T is indeed a clique tree of G, it remains to prove the "connectivity condition" for T. Namely, consider nodes $C_1, C_2 \in V(T)$ and a node C_3 on the path in T between C_1 and C_2. Since φ is an isomorphism between T and T', we have $\varphi(C_i) \in V(T')$ for $i = 1,2,3$ and $\varphi(C_3)$ lies on the path in T' between $\varphi(C_1)$ and $\varphi(C_2)$. Thus, we conclude $\varphi(C_3) \supseteq \varphi(C_1) \cap \varphi(C_2)$ because T' is a clique tree. So we write $C_3 = \varphi(C_3) \cap V(G) \supseteq \varphi(C_1) \cap \varphi(C_2) \cap V(G) = C_1 \cap C_2$. This proves (2).

We have proved in (2) that T is a clique tree of G. Notably, as T^* is also a clique tree of G, we conclude that both T and T^* have the same vertex set, i.e., $V(T) = V(T^*)$. We now look at the edges of T.

(3) $F \subseteq E(T)$

Consider an edge $e = CC' \in F$, and recall the definition of φ and the claim (1). From this it follows that $\varphi(C)$ and $\varphi(C')$ are the only maximal cliques of G' that contain v_e. As $\varphi(C)$ and $\varphi(C')$ are also nodes of T' which is a clique tree of G', we conclude that every maximal clique on the path in T' between $\varphi(C)$ and $\varphi(C')$ also contains v_e. But, as mentioned above, the vertex v_e is in no other maximal clique of G'. So this is only possible if $\varphi(C)$ and $\varphi(C')$ are adjacent in T'. Consequently, C and C' are adjacent in T, namely $e \in E(T)$. This proves (3).

(4) $\mathscr{H}(T^*) \subseteq \mathscr{H}(T)$ and each $C \in \mathscr{H}(T^*)$ satisfies $N_{T^*}(C) \subseteq N_T(C)$.

Consider $C \in \mathscr{H}(T^*)$, namely C is a node of T^* with at least three neighbours in T^*. Then, by the definition of $\mathscr{E}(T^*)$, all edges incident to C in T^* belong to $\mathscr{E}(T^*)$. As $\mathscr{E}(T^*) = F$ and $F \subseteq E(T)$ by (3), the edges incident to C in T^* are also edges of T. In other words, every neighbour of C in T^* is a neighbour of C in T, namely $N_T(C) \supseteq N_{T^*}(C)$. Thus C has at least three neighbours in T implying $C \in \mathscr{H}(T)$. This proves (4).

(5) $\mathscr{H}(T) = \mathscr{H}(T^*)$ and $\mathscr{E}(T) = \mathscr{E}(T^*)$.

By (4), we conclude $\mathscr{H}(T) \supseteq \mathscr{H}(T^*)$. Now, we calculate using (1), (4), and $(\star\star)$ as follows.

$$\ell(G') \leq |\mathscr{L}(T^*)| = 2 + \sum_{C \in \mathscr{H}(T^*)} \big(\deg_{T^*}(C) - 2\big) \leq 2 + \sum_{C \in \mathscr{H}(T)} \big(\deg_T(C) - 2\big) = |\mathscr{L}(T)| = \ell(G')$$

Note that the second inequality follows from (4) and the fact that $\deg_T(C) \geq 3$ for all $C \in \mathscr{H}(T)$, while the last equality is by $\ell(G') = |\mathscr{L}(T')|$ and the fact that T and T' are isomorphic.

Thus the inequalities in the above formula are, in fact, equalities. Therefore, using (4), we conclude that $\mathscr{H}(T) = \mathscr{H}(T^*)$ and every $C \in \mathscr{H}(T^*)$ satisfies $N_T(C) = N_{T^*}(C)$. To see this, recall that each $C \in \mathscr{H}(T^*)$ contributes to the sum on the right at least as much as to the sum on the left, since $N_T(C) \supseteq N_{T^*}(C)$ by (4). Further, every $C \in \mathscr{H}(T)$ has a positive contribution to the sum on the right as $\deg_T(C) \geq 3$ by the definition of $\mathscr{H}(T)$. Thus, since the two sums are equal, the only possibility is that $\mathscr{H}(T) = \mathscr{H}(T^*)$ and that each $C \in \mathscr{H}(T^*)$ satisfies $N_T(C) = N_{T^*}(C)$ as claimed.

To conclude the proof, recall that $\mathscr{E}(T)$, resp. $\mathscr{E}(T^*)$, is the set of edges of T, resp. T^*, incident to the nodes in $\mathscr{H}(T)$, resp. $\mathscr{H}(T^*)$. As $\mathscr{H}(T) = \mathscr{H}(T^*)$ and each $C \in \mathscr{H}(T) = \mathscr{H}(T^*)$ is incident to the same set of edges in T and T^* for it satisfies $N_T(C) = N_{T^*}(C)$, we conclude that $\mathscr{E}(T) = \mathscr{E}(T^*)$. This proves (5).

Now (5) and (2) prove that T is indeed a solution, namely that T is a clique tree of G with $\mathscr{E}(T) = \mathscr{E}(T^*) = F$. Hence, the test in Line 5 succeds and the algorithm correctly returns a solution in Line 6.

This concludes the proof of correctness of the algorithm. To address the complexity, let $n = |V(G)|$ as usual. First, we note that we may assume that F contains at most $n-1$ edges as no clique tree of G has more than n nodes. If this is not so, we can safely report that no solution exists. Thus, as G' has $|V(G)| + |F| = O(n)$ vertices, we

conclude that step 3 takes $O(n^3)$ time using the algorithm of [12]. All other steps clearly take at most $O(n^2)$ time. Notably, in step 2 we use a linear time algorithm from [20].

Thus the total complexity is $O(n^3)$ as promised. That concludes the proof. □

Finally, we are ready to prove Theorem 4.

Proof of Theorem 4. Let G be a chordal graph with $\ell(G) \leq \ell$. By Corollary 6 (proven in §5), there exists a tree model of G that simultaneously realizes both the leafage and the vertex leafage of G. By the remarks in §2, there is also a clique tree of G with this property; let T^* denote this clique tree. In other words, the tree T^* satisfies $|\mathscr{L}(T^*)| = \ell(G)$ and $|\mathscr{L}(T^*_u)| \leq v\ell(G)$ for all $u \in V(G)$ where $T^*_u = T^*\big[\{C \in V(T^*) \mid u \in C\}\big]$.

By Lemmas 8 and 9, it suffices to know the set $\mathscr{E}(T^*)$ to be able, in polynomial time, to find a tree model of G (possibly different from T^*) that realizes the vertex leafage of G. This forms the basis of our algorithm as follows.

Let $F \subseteq E(\mathscr{C}(G))$. If there exists a clique tree T with $\mathscr{E}(T) = F$, define $\alpha_F = \max_{u \in V(G)} |\mathscr{L}(T_u)|$ where $T_u = T\big[\{C \in V(T) \mid u \in C\}\big]$. If such a tree does not exist, define $\alpha_F = +\infty$. Note that the value of α_F is well-defined, since by Lemma 8 it is independent of the particular choice of the clique tree T. Thus, the value of α_F can be determined, for any given F, in time $O(n^3)$ using Lemma 9. In particular, $\alpha_{\mathscr{E}(T^*)} = v\ell(G)$ by the choice of T^*.

Our algorithm tries all possible sets $F \subseteq \mathscr{C}(G)$ of size at most $3\ell - 6$ as candidates for $\mathscr{E}(T^*)$ and chooses one that that minimizes α_F. If F_{opt} is this set, the algorithm outputs a clique tree T_{opt} of G with $\mathscr{E}(T_{opt}) = F_{opt}$.

We claim that this procedure correctly finds a clique tree of G that realizes the vertex leafage of G. By $(\star\star)$, we observe that $\mathscr{E}(T^*) \leq 3|\mathscr{L}(T^*)| - 6 \leq 3\ell - 6$. Thus, the algorithm must, at some point, consider $\mathscr{E}(T^*)$ as the set F. For this set, we have $\alpha_F = \alpha_{\mathscr{E}(T^*)} = v\ell(G)$. By the minimality of F_{opt}, we conclude $\alpha_{F_{opt}} \leq \alpha_{\mathscr{E}(T^*)} = v\ell(G)$. Hence, $\alpha_{F_{opt}} < \infty$ and so the tree T_{opt} exists. Moreover, $\alpha_F \geq v\ell(G)$ for all sets F, by the definition of $v\ell(G)$ and α_F. Thus, we must conclude $\alpha_{F_{opt}} = v\ell(G)$ and consequently by Lemma 8, the tree T_{opt} is a clique tree of G that realizes the vertex leafage of G. This proves the correctness of our algorithm.

Finally, let us analyze the complexity. Let $n = |V(G)|$ as usual. Recall that G has at most n maximal cliques. Thus there are at most n^2 edges in $\mathscr{C}(G)$, and hence, at most $n^{6\ell-12}$ choices for the set F. For each choice of F, we use Lemma 9 to find a clique tree T with $\mathscr{E}(T) = F$ if it exists. This takes $O(n^3)$ for each F, including the calculation of α_F. Altogether, the running time is $O(n^{6\ell-9}) = n^{O(\ell)}$ as promised. That concludes the proof. □

5. Vertex Leafage with Optimum Leafage

In this section, we prove Theorem 5 and Corollary 6. Namely, we demonstrate that the algorithm from [12], solving the leafage problem, satisfies the claim of Theorem 5. This algorithm, given a chordal graph G, outputs a clique tree of G with minimum possible number of leaves. This is done by starting from an arbitrary clique tree T of G, and iteratively decreasing the number of leaves of T as long as possible.

We observe (and formally prove later in this section) that this process has the additional property that it never increases the number of leaves in the subtrees of the tree model $\mathscr{T}_T$ defined by T. In other words, if T^* is the clique tree resulting from this process, then $\mathscr{T}^* = \mathscr{T}_{T^*}$ satisfies the claim of Theorem 5. This will imply that if the starting clique tree T realizes the vertex leafage of G, then $\mathscr{T}^* = \mathscr{T}_{T^*}$ satisfies the claim of Corollary 6.

For the proof of the above, we need to explain the inner workings of the algorithm from [12]. This algorithm, in place of clique trees, operates on the so-called token assignments defined as follows.

For a chordal graph G, a *token assignment* of G is a function τ that assigns to every maximal clique C of G, a multiset $\tau(C)$ of subsets of C. We use the word *token* for the members of $\tau(C)$. Note that the same subset may appear in $\tau(C)$ many times. We focus on special token assignment that arise from clique trees.

The token assignment *defined* by a clique tree T of G, and denoted by ε_T, assigns to every maximal clique C of G, the multiset $\varepsilon_T(C) = \{C \cap C' \mid CC' \in E(T)\}$. In other words, $\varepsilon_T(C)$ consists of the intersections of C with its neighbours in T. A token assignment τ is *realizable* if there is a clique tree T of G such that $\tau = \varepsilon_T$.

(See Figure 3 for an illustration of these concepts.)

Notice that the token assignment $\tau = \varepsilon_T$ contains all the information needed to determine the number of leaves in T and also the number of leaves in the subtrees of the corresponding model $\mathscr{T}_T$. We summarize this as follows.

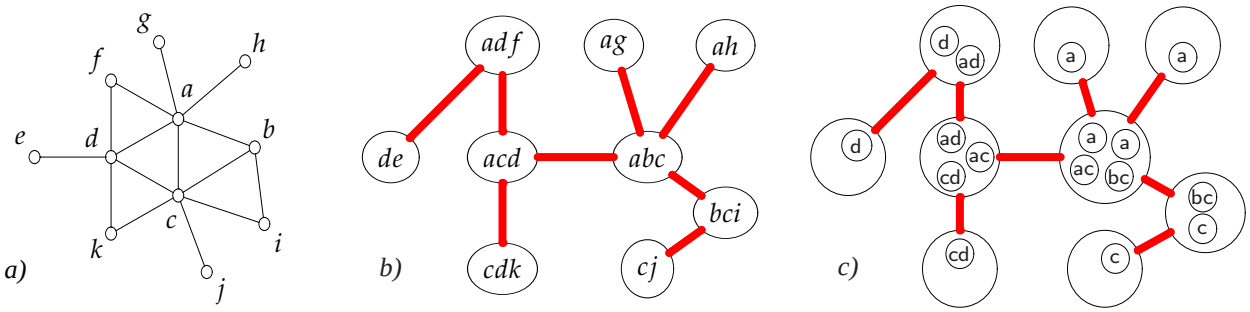

Figure 3: *a)* Example chordal graph G, *b)* clique tree T of G, *c)* token assignment $\tau = \varepsilon_T$.

Lemma 10. *Let G be a chordal graph, let T be a clique tree of G, and let $\mathscr{T}_T = \big(T, \{T_u\}_{u \in V(G)}\big)$ denote the tree model of G defined by T. Let $\tau = \varepsilon_T$, and define $\tau_u(C) = \{S \mid S \in \tau(C), u \in S\}$ for each $u \in V(G)$. Then*

- $\deg_T(C) = |\tau(C)|$ *for all* $C \in V(T)$, *and*
- $\deg_{T_u}(C) = |\tau_u(C)|$ *for all* $u \in V(G)$ *and all* $C \in V(T_u)$.

Consequently, $\mathscr{L}(T) = \Big\{C \;\Big|\; |\tau(C)| = 1\Big\}$ *and* $\mathscr{L}(T_u) = \Big\{C \;\Big|\; |\tau_u(C)| = 1\Big\}$ *for all* $u \in V(G)$.

In particular, while there can be multiple clique trees defining the same token assignment, these clique trees will have the same sets of leaves and consequently we do not need to distinguish them from one another. In other words, it suffices to maintain that the token assignment we consider corresponds to some clique tree of G. This can be tested easily by applying four particular conditions as described in [12]. As we do not use this test here directly, we omit further details. (For more, see [12, Theorem 6].)

Now, we are finally ready to explain the main steps of the algorithm from [12]. The algorithm is given a chordal graph G and a clique tree T of G. It starts by constructing the token assignment $\tau = \varepsilon_T$. Then it proceeds iteratively. During each iteration step, a current token assignment τ is examined to determine if there exists a different token assignment corresponding to a clique tree with fewer leaves. This is done by checking for an *augmenting path* in τ, which is a specific sequence of *token moves* (see definitions below). If an augmenting path exists, we pick the shortest such path and exchange tokens along the path. This results in a new token assignment τ that corresponds to a clique tree with fewer leaves. If no augmenting path exists, we arrive at an optimal solution (i.e., a token assignment whose corresponding clique trees all have $\ell(G)$ leaves) and we output this solution. We summarize the above procedure as Algorithm 2. Below we provide the missing definitions.

Let G be a chordal graph and τ be a token assignment of G. A *token move* is an ordered triple (C_1, C_2, S) where C_1, C_2 are maximal cliques of G and $S \in \tau(C_1)$. For a token move (C_1, C_2, S), we write $\tau \div (C_1, C_2, S)$ to denote the token assignment τ' that is the result of moving S from $\tau(C_1)$ to $\tau(C_2)$. Namely[1], we have $\tau'(C_1) = \tau(C_1) \setminus \{S\}$ and $\tau'(C_2) = \tau(C_2) \cup \{S\}$, while $\tau'(C) = \tau(C)$ for all other $C \notin \{C_1, C_2\}$.

A sequence of token moves (C_1, C_2, S_1), (C_2, C_3, S_2), …, (C_{k-1}, C_k, S_{k-1}) where $k > 1$ is an *augmenting path* of τ if $|\tau(C_k)| = 1$ and each $j \in \{1, \ldots, k-1\}$ satisfies

(i) $\tau \div (C_j, C_{j+1}, S_j)$ is a realizable token assignment[2], and (ii) $|\tau(C_j)| = \begin{cases} \geq 3 & \text{if } j = 1 \\ 2 & \text{otherwise} \end{cases}$

See Figure 4 for an example of an augmenting path of a token assignment τ and its application to τ.

It is easy to see that the application of an augmenting path decreases the number of leaves in the resulting token assignment. This, however, does not guarantee that the resulting assignment corresponds to a clique tree of G. Fortunately, it can be proved that a shortest augmenting path has this property, and moreover, there always exists an augmenting path unless τ corresponds to an optimal clique tree. The details can be found in [12]. We only remark the following invariant which is maintaned throughout the algorithm.

[1]Note that as both $\tau(C_1)$ and $\tau'(C_1)$ are multisets, to obtain $\tau'(C_1)$ we only remove one instance of S from $\tau(C_1)$ in case S appears in $\tau(C_1)$ several times. This is consistent with the semantics of the set difference for multisets.

[2]i.e., it corresponds to a clique tree of G.

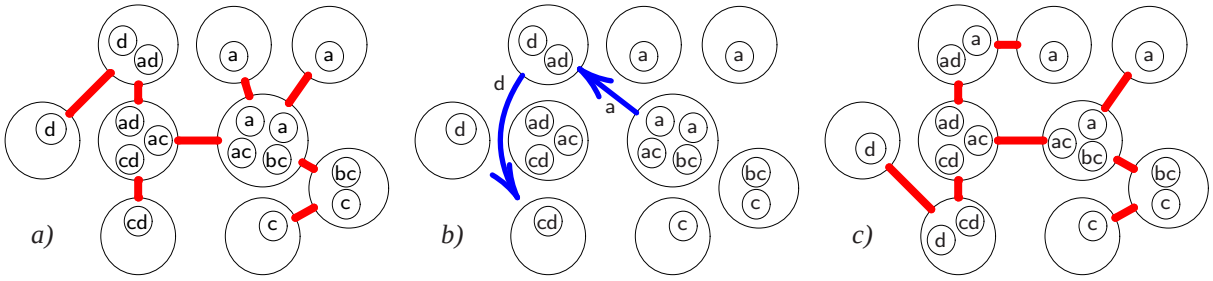

Figure 4: *a)* token assignment τ (see Figure 3), *b)* augmenting path (abc, adf, a), (adf, cdk, d) – directed edges, *c)* τ after applying the path.

Algorithm 2: *Leafage(G,T)*

Input: A chordal graph G, and a clique tree T of G.
Output: A clique tree T^* of G with $|\mathscr{L}(T^*)| = \ell(G)$.

1 Initialize $\tau \leftarrow \varepsilon_T$ /* *initialize the token assignment with the given clique tree.* */
2 **while** there exists an augmenting path of τ **do**
3 Let $(C_1, C_2, S_1), \ldots, (C_{k-1}, C_k, S_{k-1})$ be a shortest augmenting path of τ
4 **for** all i from 1 to $k-1$ **do**
5 $\tau \leftarrow \tau \div (C_i, C_{i+1}, S_i)$
6 **return** T^* where $\varepsilon_{T^*} = \tau$.

Lemma 11. [12] *In Line 2 of Algorithm 2, the variable τ is a realizable token assignment.*

After this introduction, we are ready to prove Theorem 5.

Proof of Theorem 5. We prove the theorem by showing that each application of an augmenting path in Algorithm 2 does not increase the number of leaves in the subtrees of the corresponding tree model.

In other words, let τ be the token assignment considered at the start of some iteration (Lines 2-5) of Algorithm 2, and let $(C_1, C_2, S_1), \ldots, (C_{k-1}, C_k, S_{k-1})$ be the shortest augmenting path of τ considered in this iteration (Line 3). Let τ' denote the value of τ after applying the token moves of this path (Lines 4-5).

By Lemma 11, both τ and τ' are realizable token assignments of G. In other words, there exist clique trees T and T' of G such that $\tau = \varepsilon_T$ and $\tau' = \varepsilon_{T'}$. Let $\mathscr{T}_T = \big(T, \{T_u\}_{u \in V(G)}\big)$ and $\mathscr{T}_{T'} = \big(T', \{T'_u\}_{u \in V(G)}\big)$ be the corresponding tree models of G. In other words, for each $u \in V(G)$, we have $T_u = T\big[\{C \in V(T) \mid u \in C\}\big]$ and $T'_u = T'\big[\{C \in V(T') \mid u \in C\}\big]$. In addition, for each $u \in V(G)$ and each maximal clique C of G, define the multisets $\tau_u(C) = \{S \mid S \in \tau(C), u \in S\}$ and $\tau'_u(C) = \{S \mid S \in \tau'(C), u \in S\}$.

Now, to prove the theorem, it suffices to demonstrate that $|\mathscr{L}(T_u)| \geq |\mathscr{L}(T'_u)|$ for every $u \in V(G)$. Consider $u \in V(G)$ and define two sequences of integers $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ where $a_i = |\tau_u(C_i)|$ and $b_i = |\tau'_u(C_i)|$ for all $i \in \{1, \ldots, k\}$. Note that $\tau_u(C) = \tau'_u(C)$ for all $C \notin \{C_1, \ldots, C_k\}$, and by Lemma 10, we have $\mathscr{L}(T_u) = \big\{C \;\big|\; |\tau_u(C)| = 1\big\}$ and $\mathscr{L}(T'_u) = \big\{C \;\big|\; |\tau'_u(C)| = 1\big\}$. This implies the following.

$$|\mathscr{L}(T_u)| - |\mathscr{L}(T'_u)| = \Big|\Big\{C_i \;\Big|\; |\tau_u(C_i)| = 1\Big\}\Big| - \Big|\Big\{C_i \;\Big|\; |\tau'_u(C_i)| = 1\Big\}\Big| = \Big|\{i \mid a_i = 1\}\Big| - \Big|\{i \mid b_i = 1\}\Big|$$

In other words, the proof boils down to showing that $\{i \mid b_i = 1\}$ does not have more elements than $\{i \mid a_i = 1\}$.

Recall that, by the definition of the augmenting path, $|\tau(C_k)| = 1$ and $|\tau(C_i)| = 2$ for all $i \in \{2, \ldots, k-1\}$. Notably, since the path is shortest, $C_1, \ldots, C_k$ are distinct maximal cliques of G. Thus, as $\tau_u(C) \subseteq \tau(C)$ for all C, we conclude that $a_k \leq 1$ and $a_i \leq 2$ for all $i \in \{2, \ldots, k-1\}$. Further, note that $|\tau'(C_k)| = 2$ while $|\tau'(C_i)| = |\tau(C_i)| = 2$ for all $i \in \{2, \ldots, k-1\}$. In other words, we have $b_i \leq 2$ for all $i \in \{2, \ldots, k\}$.

We shall use the following two claims to show that $\big|\{i \mid b_i = 1\}\big| \leq \big|\{i \mid a_i = 1\}\big|$.

(6) If $b_i = 1$, then $a_i \geq 1$.

Consider $i \in \{1,\ldots,k\}$ such that $b_i = 1$, and assume for contradiction that $a_i = 0$. Since $b_i = 1$, we have by Lemma 10 that $1 = b_i = |\tau'_u(C_i)| = \deg_{T'_u}(C_i)$. In other words, C_i is a leaf of T'_u, and thus T'_u contains at least two vertices. Recall that $V(T_u) = V(T'_u)$, and note that $0 = a_i = |\tau_u(C_i)| = \deg_{T_u}(C_i)$ by Lemma 10. This means that C_i is a vertex of T_u with no neighbour in T_u. This is clearly impossible, since T_u is connected and $|V(T_u)| = |V(T'_u)| \geq 2$. Thus we must conclude that $a_i \geq 1$. This proves (6).

(7) If $b_i = 1$ and $a_i \geq 2$, then there exists $j > i$ such that $a_j = 1$, $b_j = 2$, and $a_r = b_r$ for all $r \in \{i+1,\ldots,j-1\}$.

To see this, first recall the construction of τ' from τ by moving the tokens $S_1,\ldots,S_{k-1}$ as follows.

$$\tau'(C_i) = \begin{cases} \tau(C_i) \setminus \{S_i\} & \text{if } i = 1 \\ \big(\tau(C_i) \setminus \{S_i\}\big) \cup \{S_{i-1}\} & \text{if } 1 < i < k \\ \tau(C_i) \cup \{S_{i-1}\} & \text{if } i = k \end{cases}$$

Also recall that $a_i = \big|\tau_u(C_i)\big| = \big|\{S \mid S \in \tau(C_i), u \in S\}\big|$ and $b_i = \big|\tau'_u(C_i)\big| = \big|\{S \mid S \in \tau'(C_i), u \in S\}\big|$. From these two facts we conclude the following relationship between the values of a_i and b_i ($1 < i < k$).

$$(\star\star\star) \quad b_1 = \begin{cases} a_1 - 1 & \text{if } u \in S_1 \\ a_1 & \text{if } u \notin S_1 \end{cases} \qquad b_i = \begin{cases} a_i & \text{if } u \in S_i \cap S_{i-1} \\ a_i - 1 & \text{if } u \in S_i \setminus S_{i-1} \\ a_i + 1 & \text{if } u \in S_{i-1} \setminus S_i \\ a_i & \text{if } u \notin S_{i-1} \cup S_i \end{cases} \qquad b_k = \begin{cases} a_k + 1 & \text{if } u \in S_{k-1} \\ a_k & \text{if } u \notin S_{k-1} \end{cases}$$

Now, for the proof of (7), consider $i \in \{1,\ldots,k\}$ such that $b_i = 1$ and $a_i \geq 2$. By $(\star\star\star)$, we have $|b_i - a_i| \leq 1$ and thus $a_i = 2$. Further, $i < k$ since $b_k \geq a_k$ by $(\star\star\star)$, but $b_i = 1 < 2 = a_i$. Moreover, $u \in S_i$ by $(\star\star\star)$, since $i < k$ and $b_i = a_i - 1$. We let j be the largest in $\{i+1,\ldots,k+1\}$ such that $a_r = b_r$ for each $r \in \{i+1,\ldots,j-1\}$.

First, we observe that $u \in S_r$ for each $r \in \{i,\ldots,j-2\}$. Indeed, if otherwise, we let r be the smallest index in $\{i,\ldots,j-2\}$ with $u \notin S_r$. As we just argued, we have $u \in S_i$, and so $r > i$. Therefore, $u \in S_{r-1}$ by the minimality of r. But then $b_r = a_r + 1$ by $(\star\star\star)$, since $1 \leq i < r < j-1 \leq k$, a contradiction.

This also implies that $j \leq k$. Indeed, if $j = k+1$, then $i \leq j-2 = k-1$ since $i < k$. Thus $u \in S_{j-2} = S_{k-1}$ which yields $b_k = a_k + 1$ by $(\star\star\star)$. However, $k \in \{i+1,\ldots,j-1\}$ and so $b_k = a_k$ by the choice of j.

We can now conclude that $u \in S_{j-1}$. Indeed, if $i = j-1$, then we use the fact that $u \in S_i$. Otherwise, $i \leq j-2$ in which case $u \in S_{j-2}$ as argued above, and thus $u \in S_{j-1}$ by $(\star\star\star)$, since $a_{j-1} = b_{j-1}$ and $1 \leq i < j-1 < k$.

Finally, we consider the value of j. First, suppose that $j = k$. Then $b_k = a_k + 1$, since $u \in S_{j-1} = S_{k-1}$. We recall that $a_k \leq 1$ and so $b_k \in \{1,2\}$. If $b_k = 1$, we have $a_k \geq 1$ by (6), but then $a_k \geq b_k = a_k + 1 > a_k$, a contradiction. So, we must conclude $b_k = 2$ and $a_k = 1$. Thus, as $j = k$, we have $b_j = 2$, $a_j = 1$, and $a_r = b_r$ for all $r \in \{i+1,\ldots,j-1\}$ as required. Thus we may assume that $j < k$. By the maximality of j, we have $a_j \neq b_j$. Also, $u \in S_{j-1}$ and $1 \leq i < j < k$. So by $(\star\star\star)$ we conclude that $b_j = a_j + 1$. We recall that $b_j \leq 2$ as $j > 1$. Thus $b_j \in \{1,2\}$ as $a_j \geq 0$. Again, if $b_j = 1$, we conclude $a_j \geq 1$ by (6) in which case $a_j \geq b_j > a_j$, a contradiction. Thus $b_j = 2$, $a_j = 1$, and $a_r = b_r$ for all $r \in \{i+1,\ldots,j-1\}$, as required. This proves (7).

We are now ready to conclude the proof. Denote $A = \{i \mid a_i = 1\}$ and $B = \{i \mid b_i = 1\}$. We show that $|B| \leq |A|$ which will imply the present theorem as argued above the claim (6).

For each $i \in B$, if $a_i = 1$, define $\varphi(i) = i$. Otherwise, define $\varphi(i) = j$ where j is the index obtained by applying (7) for i. Note that $a_j = 1$ and $b_j = 2$. It follows that φ is a mapping from B to A.

We show that φ is, in fact, an injective mapping. Suppose otherwise, and let i, i^+ be distinct elements of B such that $\varphi(i) = \varphi(i^+)$. Recall that $b_i = b_{i^+} = 1$ and note that $i \leq \varphi(i)$ and $i^+ \leq \varphi(i^+)$. If $i = \varphi(i)$, then $i^+ \leq \varphi(i^+) = \varphi(i) = i$ implying $i^+ < \varphi(i^+)$ as i and i^+ are distinct. So $a_{i^+} \neq 1$ by the definition of φ, and hence $b_{\varphi(i^+)} = 2$ as $\varphi(i^+)$ was obtained by applying (7) for i^+. But then $1 = b_i = b_{\varphi(i)} = b_{\varphi(i^+)} = 2$, a contradiction. Thus we must conclude that $i < \varphi(i)$ and, by symmetry, also $i^+ < \varphi(i^+)$. Now, without loss of generality, assume $i < i^+$. Since $i^+ < \varphi(i^+)$, we must have $a_{i^+} \neq 1$ by the definition of φ. However, $b_{i^+} = 1$ as $i^+ \in B$, and hence, $a_{i^+} \neq b_{i^+}$. Recall that the choice of $\varphi(i)$ using (7) for i guarantees that $a_r = b_r$ for all $r \in \{i+1,\ldots,\varphi(i)-1\}$.

In particular, $i < i^+ < \varphi(i^+) = \varphi(i)$ and so $a_{i^+} = b_{i^+}$ which is a contradition. This verifies that φ is indeed an injective mapping from B to A, which yields $|B| \leq |A|$.

This completes the proof of Theorem 5. □

6. Concluding Remarks

In this paper we have studied the vertex leafage of chordal graphs. Specifically, a chordal graph $G = (V, E)$ has vertex leafage k when it has a tree model $\left(T, \{T_u\}_{u \in V}\right)$ such that each subtree T_u has at most k leaves.

We have shown that, for every fixed $k \geq 3$, it is NP-complete to decide if a split graph G has vertex leafage at most k, even when G is known to have vertex leafage at most $k+1$. Moreover, we have proved that it is NP-hard to find a tree model of G with as few leaves in subtrees in total as possible, and it is also NP-hard to find a clique tree where as many subtrees as possible are paths, even if G is a split graph of vertex leafage 3. Interestingly, this makes the polynomial-time recognition of path graphs [2, 8, 18] the only tractable unparameterized case of this problem.

On the positive side, we have demonstrated an $n^{O(\ell)}$ algorithm to compute the vertex leafage of a chordal graph whose leafage is bounded by ℓ. This puts vertex leafage in the class XP when the leafage is taken as the parameter.

Finally, we have shown that every chordal graph G has a tree model which simultaneously realizes G's leafage and vertex leafage. In proving this result we have also shown that, for every path graph G, there exists a path model with $\ell(G)$ leaves in the host tree and such a path model can be computed in $O(n^3)$ time.

The following questions remain open.

(A) for parameter p (integer) is any of the following problems in XP:
  - given a chordal graph G is there a tree model of G where at most p subtrees are paths?
  - given a chordal graph G is there a tree model of G where the total number of leaves in all subtrees is at most p?

(B) if the answer to (A) is affirmative, is the problem in question in FPT or is it W[t]-hard for some (all) t?

(C) is the vertex leafage FPT with respect to leafage?

(D) is the vertex leafage FPT with respect to some other graph parameter?

Acknowledgement

We would like to thank anonymous referees for thoroughly reading the manuscript and for providing helpful comments. We would also like to acknowledge Marisa Gutierrez and Pablo de Caria whose work [4] and later discussions inspired the proofs of Theorems 3 and 4.

The initial work on this project was done during the second author's visit at the University of Toronto in 2009 and later during the first author's visit at the Caesarea Rothschild Institute of the University of Haifa in 2011. Both trips were made possible by a generous support of Prof. Derek Corneil of the University of Toronto via his NSERC grant. The second author also gratefully acknowledges support from EPSRC, award EP/I01795X/1.

References

[1] BUNEMAN, P. A characterization of rigid circuit graphs. *Discrete Mathematics 9* (1974), 205–212.

[2] CHAPLICK, S. *Path Graphs and PR-trees.* PhD thesis, University of Toronto, 2012.

[3] COHEN, E., GOLUMBIC, M., LIPSHTEYN, M., AND STERN, M. What is between chordal and weakly chordal graphs? In *Graph-Theoretic Concepts in Computer Science (WG 2008), Lecture Notes in Computer Science 5344* (2008), pp. 275–286.

[4] DE CARIA, P., AND GUTIERREZ, M. Determining possible sets of leaves for spanning trees of dually chordal graphs. In *MACI 2009: Congreso de Matemática Aplicada, Computacional e Industrial* (2009), pp. 153–156.

[5] FULKERSON, D. R., AND GROSS, O. A. Incidence matrices and interval graphs. *Pacific Journal of Mathematics 15* (1965), 835–855.

[6] GAREY, M. R., AND JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[7] GAVRIL, F. The intersection graphs of subtrees of trees are exactly the chordal graphs. *Journal of Combinatorial Theory Series B 16* (1974), 47–56.

[8] GAVRIL, F. A recognition algorithm for the intersection graphs of paths in trees. *Discrete Mathematics 23* (1978), 211–227.

[9] GAVRIL, F. Generating the maximum spanning trees of a weighted graph. *Journal of Algorithms 8* (1987), 592–597.

[10] GOLUMBIC, M. C., LIPSHTEYN, M., AND STERN, M. Equivalences and the complete hierarchy of intersection graphs of paths in a tree. *Discrete Applied Mathematics 156* (2008), 3203–3215.

[11] HABIB, M., AND STACHO, J. Linear algorithms for chordal graphs of bounded directed vertex leafage. *Electronic Notes in Discrete Mathematics 32* (2009), 99–108.
[12] HABIB, M., AND STACHO, J. Polynomial-time algorithm for the leafage of chordal graphs. In *Algorithms - ESA 2009, Lecture Notes in Computer Science 5757* (2009), pp. 290–300.
[13] JAMISON, R. E., AND MULDER, H. M. Constant tolerance intersection graphs of subtrees of a tree. *Discrete Mathematics 290* (2005), 27–46.
[14] LÉVÊQUE, B., MAFFRAY, F., AND PREISSMANN, M. Characterizing path graphs by forbidden induced subgraphs. *Journal of Graph Theory 62* (2009), 369–384.
[15] LIN, I. J., MCKEE, T. A., AND WEST, D. B. The leafage of a chordal graph. *Discuss. Math. Graph Theory 18* (1998), 23–48.
[16] MONMA, C. L., AND WEI, V. K.-W. Intersection graphs of paths in a tree. *Jounal of Combinatorial Theory B 41* (1986), 141–181.
[17] SCHAEFER, T. J. The complexity of satisfiability problems. In *STOC* (1978), pp. 216–226.
[18] SCHÄFFER, A. A. A faster algorithm to recognize undirected path graphs. *Discrete Applied Mathematics 43* (1993), 261–295.
[19] SHIBATA, Y. On the tree representation of chordal graphs. *Journal of Graph Theory 12* (1988), 421–428.
[20] TARJAN, R. E. Depth first search and linear graph algorithms. *SIAM Journal on Computing 1* (1972), 146–160.
[21] WALTER, J. R. Representations of chordal graphs as subtrees of a tree. *Journal of Graph Theory 2* (1978), 265–267.