Polynomial-time algorithm for the leafage of chordal graphs

Michel Habib and Juraj Stacho

LIAFA – CNRS and Université Paris Diderot – Paris VII,
Case 7014, 75205 Paris Cedex 13, France
{habib,jstacho}@liafa.jussieu.fr

Abstract. Every chordal graph G can be represented as the intersection graph of a collection of subtrees of a host tree, the so-called tree model of G. The leafage $l(G)$ of a connected chordal graph G is the minimum number of leaves of the host tree of a tree model of G. This concept was first defined by I.-J. Lin, T.A. McKee, and D.B. West in [9]. In this contribution, we present the first polynomial time algorithm for computing $l(G)$ for a given chordal graph G. In fact, our algorithm runs in time $O(n^3)$ and it also constructs a tree model of G whose host tree has $l(G)$ leaves.

1 Introduction

In this paper, graph is always simple, undirected and loopless.

A graph is *chordal*, if it has no induced cycles of length four or longer. By a result of Gavril [5], a graph G is chordal if and only if G can be represented as the intersection graph of a collection of subtrees of a host tree, the so-called *tree model* of G. The *leafage* $l(G)$ of a connected chordal graph G is defined as the minimum number of leaves of the host tree of a tree model of G. If G is an interval graph (the intersection graph of intervals of the real line), we always have $l(G) = 2$. Hence, the leafage can be seen as a measure of how far a chordal graph is from being an interval graph. This has several algorithmic consequences. For instance, it is shown in [7] that a chordal graph with bounded leafage always has a so-called implicit representation which allows some problems on such graphs to be solved more efficiently. Moreover, efficient solutions to NP-hard problems on interval graphs naturally extend to efficient solutions on chordal graphs whose leafage is bounded; e.g., the k-subcolouring problem [2, 12, 13].

The leafage of chordal graphs was first introduced by I.-J. Lin, T.A. McKee, and D.B. West in [9] where the authors establish several bounds on this parameter for special cases of chordal graphs such as block graphs, split graphs, and k-trees. Their bounds imply polynomial time algorithms for computing the leafage in some of these special cases; however, the general question of complexity of computing $l(G)$ for a given chordal graph G

is not addressed. Since their paper, this question remained unresolved [14] except in special cases such as split graphs [8, 9] and the case of deciding $l(G) \leq k$ for $k \in \{2, 3\}$; the case $k = 2$ corresponds to interval graph recognition which is polynomial [1], and $k = 3$ is polynomial by [10].

In this paper, we finally resolve this question by providing a polynomial time algorithm computing $l(G)$ for any given chordal graph G. In particular, our algorithm runs in time $O(n^3)$ where n is the number of vertices of G and also outputs a tree model of G with $l(G)$ leaves.

The paper is structured as follows. In Section 2, we define basic notions such as clique tree and the reduced clique graph, and we show some of their useful properties related to the leafage. In Section 3, we introduce the so-called token mappings and explain their relationship to clique trees. In Section 4, we define augmenting paths and sequences and explain how they can be used to decrease the number of leaves in a given clique tree. Finally, in Section 5, we describe our algorithm and analyze its complexity.

2 Basic concepts

For a graph G and a set $X \subseteq V(G)$, we denote by $G[X]$ the subgraph of G induced on X, and denote by $G - X$ the subgraph $G[V(G) \setminus X]$. A *complete subgraph* or *clique* of G is a (not necessarily maximal) set of pairwise adjacent vertices of G. (For a complete terminology, see [15].)

Let G be a connected chordal graph. A *clique tree* of G is any tree T whose vertices are the maximal cliques of G such that for every two maximal cliques C, C', each clique on the path from C to C' in T contains $C \cap C'$. We shall assume that every edge CC' of T is labeled by $C \cap C'$. (See Figure 1 for an example of a clique tree.)

Any clique tree T can be seen as a tree model of G whose host tree is T. It is shown in [9] that G always has a clique tree with $l(G)$ leaves. Hence, in the rest of the paper, we shall focus on clique trees.

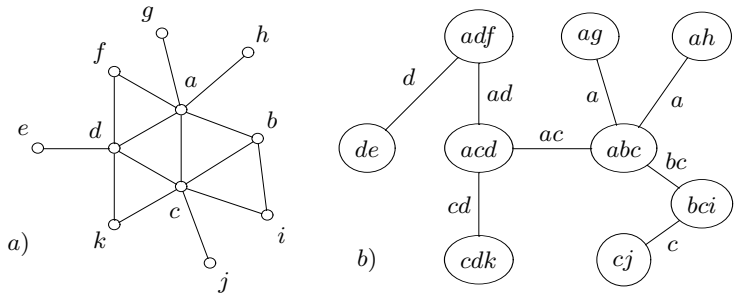

Fig. 1. *a)* Example chordal graph G, *b)* a clique tree T of G.

Cliques C, C' of G form a *separating pair*, if every path from a vertex of $C \setminus C'$ to a vertex of $C' \setminus C$ contains a vertex of $C \cap C'$. The *reduced clique graph* $\mathcal{C}_r(G)$ of G is a graph whose vertices are the maximal cliques of G, and whose edges CC' are between cliques C, C' forming separating pairs. In addition, each edge CC' of $\mathcal{C}_r(G)$ is labeled by $C \cap C'$.

The following is a fundamental result about reduced clique graphs.

Theorem 1. [4] *A tree T is a clique tree of G if and only if T is a maximum weight spanning tree of $\mathcal{C}_r(G)$ where the weight of each edge CC' is defined as $|C \cap C'|$. Moreover, the reduced clique graph $\mathcal{C}_r(G)$ is precisely the union of all clique trees of G.*

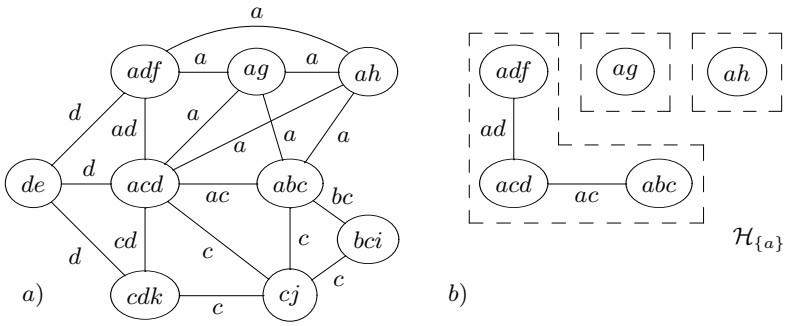

Fig. 2. *a)* The reduced clique graph $\mathcal{C}_r(G)$ of G, *b)* minimal separator graph $\mathcal{H}_{\{a\}}$

2.1 Minimal separator graphs

Let a, b be two vertices of G. A subset S of the vertices of G *disconnects* a from b in G, if a and b are in different connected components of $G - S$.

A subset S of the vertices of G is called a *minimal separator*, if there exist vertices a and b such that (i) S disconnects a from b, and (ii) no proper subset of S disconnects a from b in G.

For each minimal separator S, let $\mathcal{R}_S$ denote the set of all maximal cliques C of G with $S \subseteq C$, and let $\mathcal{H}_S$ denote the graph whose vertex set is $\mathcal{R}_S$ and whose edges are between cliques C, C' such that $C \cap C' \supsetneq S$.

Using the graphs $\mathcal{H}_S$ we can characterize the structure of $\mathcal{C}_r(G)$. (The proof is omitted due to the length restriction.)

Theorem 2. *CC' is an edge of $\mathcal{C}_r(G)$ with label $S = C \cap C'$ if and only if C and C' belong to different connected components of $\mathcal{H}_S$.* □

Let $\mathcal{S}(G)$ denote the set of all minimal separators of G. Note that chordality of G implies that for every $S \in \mathcal{S}(G)$, there exist cliques C, C' that form a separating pair such that $C \cap C' = S$. This implies, in particular, that every minimal separator of G appears on some edge of $\mathcal{C}_r(G)$.

2.2 Structure of clique trees

Let T be a clique tree of G. By Theorem 1, we have that T is characterized in terms of $\mathcal{C}_r(G)$ as a maximum weight spanning tree of $\mathcal{C}_r(G)$. Conversely, we can characterize the reduced clique graph $\mathcal{C}_r(G)$ in terms of T as follows. (The proof is omitted due to the length restriction.)

Theorem 3. *CC' is an edge of $\mathcal{C}_r(G)$ with label $S = C \cap C'$ if and only if there exists an edge with label S on the path from C to C' in T.* □

Furthermore, we can describe the structure of T in terms of the graphs $\mathcal{H}_S$.

Theorem 4. *Let $S \in \mathcal{S}(G)$ and let k_S denote the number of connected components of $\mathcal{H}_S$. Then T contains exactly $(k_S - 1)$ edges with label S, and each connected component of $\mathcal{H}_S$ induces a connected subgraph in T.*

Proof. Let C, C' be two vertices of $\mathcal{H}_S$, and let P be the path from C to C' in T. Clearly, every vertex on P must belong to $\mathcal{H}_S$ because it contains $C \cap C' \supseteq S$ by the definition of clique tree. Hence, the vertices of $\mathcal{H}_S$ induce a connected subgraph in T. Next, suppose that C, C' belong to some component $\mathcal{K}$ of $\mathcal{H}_S$. Again, it follows that every vertex of P also belongs to $\mathcal{K}$ (details omitted). Hence, both the vertices of $\mathcal{H}_S$ and of each component of $\mathcal{H}_S$ induce a subtree in T. The claim now follows. □

Note that T has at most n vertices by [3], and for each minimal separator S of G, we have $k_S \geq 2$ since there is at least one edge with label S in $\mathcal{C}_r(G)$ (see the remark above). Hence, every minimal separator of G appears on some edge of T, and conversely, for every edge CC' of T, the set $C \cap C'$ is a minimal separator and C, C' form a separating pair (by Theorem 1). In particular, G has at most $n - 1$ minimal separators.

3 Degrees and Tokens

A *degree mapping* assigns to each vertex of a graph its degree.

Theorem 5. [15] *A mapping $f : X \to \mathbb{N}$ is a degree mapping of a tree if and only if*

(i) $1 \leq f(x) \leq |X| - 1$ *for each* $x \in X$, *and*
(ii) $\sum_{x \in X} f(x) = 2|X| - 2$.

We define a similar notion. A *token mapping* is a mapping τ that assigns to each maximal clique C of G a distinct set of *tokens* $\tau(C)$ where

each token is labeled by some minimal separator of G. If a token t belongs to $\tau(C)$, we also say that t *is a token of* τ and that t *belongs to* C *in* τ.

Let T be a clique tree of G. The *extended degree mapping* of T, denoted by ε_T, is a token mapping that assigns to each maximal clique C a set $\varepsilon_T(C)$ of tokens corresponding to the edges incident to C in T where each token is labeled by the label of the corresponding edge. (See Figure 3 for an example of a clique tree T and its extended degree mapping ε_T.)

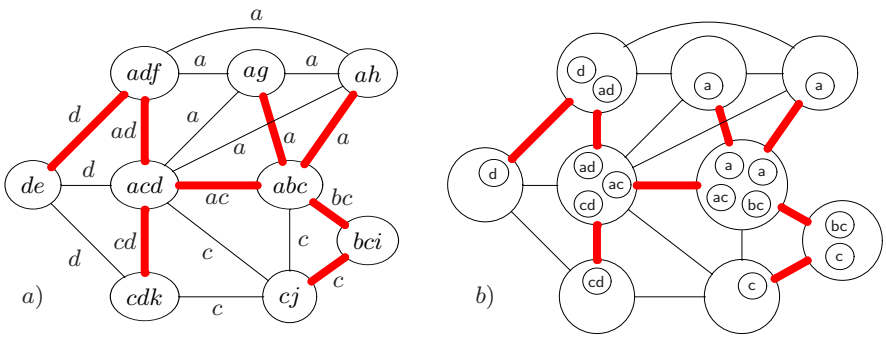

Fig. 3. *a)* the graph $\mathcal{C}_r(G)$ with an example clique tree T, *b)* token mapping $\tau = \varepsilon_T$.

Using Theorem 4, we now describe necessary and sufficient conditions characterizing extended degree mappings of clique trees.

Theorem 6. *A token mapping τ is an extended degree mapping of a clique tree of G if and only if*

(R1) *for each maximal clique C of G, the set $\tau(C)$ is non-empty,*
(R2) *for each minimal separator S of G, if a token of τ with label S belongs to $\tau(C)$ for some maximal clique C of G, then $C \supseteq S$,*
(R3) *for each minimal separator S of G, the number of tokens of τ with label S is exactly $2k_S - 2$ where k_S is the number of components of $\mathcal{H}_S$,*
(R4) *for each minimal separator S of G and every component $\mathcal{K}$ of $\mathcal{H}_S$, there exists a token with label S in $\tau(C)$ for some vertex C of $\mathcal{K}$.*

Proof. The forward direction follows directly from Theorem 4. For the backward direction, let τ be a token mapping satisfying (R1-R4). We construct a subgraph T of $\mathcal{C}_r(G)$ as follows. The vertices of T are the maximal cliques of G, and initially T has no edges.

We consider every minimal separator S of G one by one as follows. We let $\mathcal{K}_1, \ldots, \mathcal{K}_{k_S}$ be the connected components of $\mathcal{H}_S$, and we let $a_1, \ldots, a_{k_S}$ denote the number of tokens of τ with label S in $\mathcal{K}_1, \ldots, \mathcal{K}_{k_S}$, respectively. By (R4), $a_i \geq 1$ for each $1 \leq i \leq k_S$, and by (R2) and (R3), we have

$\sum_{i=1}^{k_S} a_i = 2k_S - 2$. Hence, by Theorem 5, there exists a tree $\mathcal{T}$ whose vertices are $\mathcal{K}_1, \ldots \mathcal{K}_S$ such that the degree of $\mathcal{K}_i$ in $\mathcal{T}$ is exactly a_i.

Now, for each $1 \leq i \leq k_S$, we let $t^i_1, \ldots, t^i_{a_i}$ be the tokens of τ on the vertices of $\mathcal{K}_i$ and let $C^i_1, \ldots, C^i_{a_i}$ be the vertices of $\mathcal{K}_i$ that contain tokens $t^i_1, \ldots, t^i_{a_i}$, respectively. (Possibly, $C^i_j = C^i_{j'}$ for some $j \neq j'$.) Also, we let $e^i_1, \ldots, e^i_{a_i}$ be a fixed ordering of edges of $\mathcal{T}$ incident to $\mathcal{K}_i$. Finally, for each edge $e = K_i K_{i'}$ of $\mathcal{T}$, we have j, j' such that $e = e^i_j = e^{i'}_{j'}$ by the above definition, and we add the edge $C^i_j C^{i'}_{j'}$ to T. (See example in Figure 4.)

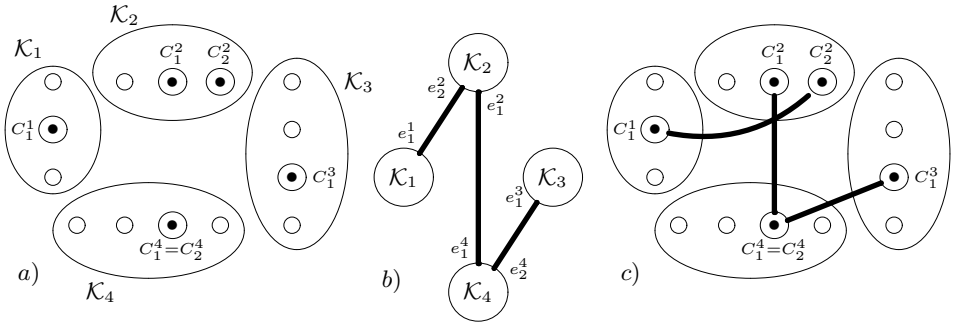

Fig. 4. *a)* Components of $\mathcal{H}_S$ and tokens with label S, *b)* The tree $\mathcal{T}$ with orderings of incident edges, *c)* edges added to T

We now show that T is a clique tree of G. First, we observe that T is a subgraph of $\mathcal{C}_r(G)$ by Theorem 2, and it contains exactly $k_S - 1$ edges with label S for every minimal separator S of G by our construction. By Theorem 4, every clique tree of G also has this property. Therefore, the weight of T is the same as the weight of any clique tree of G. By Theorem 1, it now suffices to show that T is connected. This can be proved by showing that $T[\mathcal{R}_S]$ is connected for each $S \in \mathcal{S}(G)$ which follows by induction on $n - |S|$. (We omit further details.) □

We say that a token mapping τ is *realizable*, if it satisfies the conditions (R1-R4). In light of the above theorem, we define the *degree* of C in τ, denoted by $\mathsf{deg}_\tau(C)$, to be the value $\mathsf{deg}_\tau(C) = |\tau(C)|$. If $\mathsf{deg}_\tau(C) = 1$, we call C a *leaf* of τ. We denote by $\#\mathsf{leaves}(\tau)$ the number of leaves of τ.

4 Alternation and augmentation

In this section, we show how to obtain from a realizable token mapping τ another realizable token mapping τ' with less number of leaves (if possible). We do this by moving tokens along certain paths on the maximal cliques; these paths are obtained by exploring an auxiliary graph $\mathcal{D}_\tau$

(described below). In particular, this process will resemble the classical maximum matching algorithm. For this reason, we shall use "alternating" and "augmenting" to describe similar notions in our algorithm.

Let $\mathcal{D}(G)$ denote the multidigraph[1] on the maximal cliques of G with labeled arcs such that $e = CC'$ is an arc of $\mathcal{D}(G)$ labeled with S if and only if C, C' belong to $\mathcal{R}_S$ where S is a minimal separator of G.

Let $e = CC'$ be an arc of $\mathcal{D}(G)$ with label S and τ be a token mapping such that $\tau(C)$ contains a token t labeled with S. We write $\tau \div e$ for the token mapping obtained from τ by removing the token t from $\tau(C)$ and adding t to $\tau(C')$.

4.1 Realizable arcs

Let τ be a realizable token mapping, and let $\mathcal{D}_\tau$ denote the digraph on the maximal cliques of G with arcs e for which $\tau \div e$ is realizable.

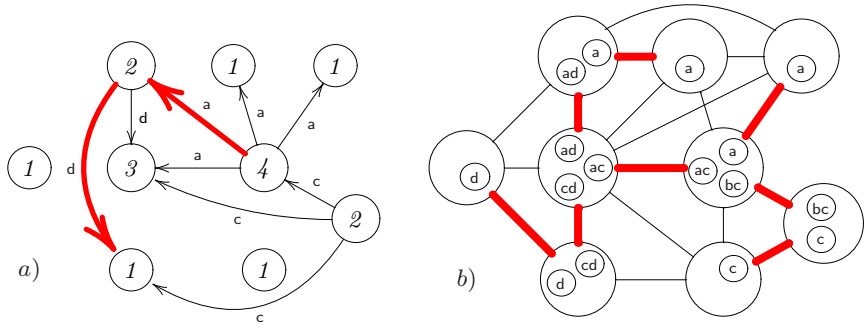

Fig. 5. *a)* the digraph $\mathcal{D}_\tau$ with an augmenting path $P = e_1, e_2$, *b)* the token mapping $\tau' = \tau \div e_1 \div e_2$ and the corresponding clique tree.

We characterize the digraph $\mathcal{D}_\tau$ as follows.

Proposition 7. *An arc $e = CC'$ with label S belongs to $\mathcal{D}_\tau$ if and only if*

(i) *C and C' are both vertices of $\mathcal{H}_S$,*
(ii) *$\tau(C)$ contains a token with label S,*
(iii) *$\mathsf{deg}_\tau(C) \geq 2$, and*
(iv) *if $\mathcal{K}$ is the connected component of $\mathcal{H}_S$ that contains C, then*
  (a) *either C' belongs to $\mathcal{K}$,*
  (b) *or C' does not belong to $\mathcal{K}$ and there exists $C'' \neq C$ in $\mathcal{K}$ such that $\tau(C'')$ contain a token with label S.*

Proof. The claim follows directly from (R1-R4). □

[1] digraph with multiple arcs between any two points allowed

4.2 Sequences

Again, let τ be a realizable token mapping.

We say that a sequence of arcs $e_1, \ldots, e_k$ of $\mathcal{D}(G)$ is a *τ-sequence*, if there exists a sequence of token mappings $\tau_0, \tau_1, \ldots, \tau_k$ where $\tau_0 = \tau$ such that for each $i \in \{1 \ldots k\}$, the arc $e_i = C_i C_i'$ satisfies

(S1) e_i is an arc of $D_{\tau_{i-1}}$,
(S2) $\tau_i = \tau_{i-1} \div e_i$,
(S3) $\mathsf{deg}_{\tau_{i-1}}(C_i) \geq 3$, and
(S4) if $i < k$ then $\mathsf{deg}_{\tau_{i-1}}(C_i') \geq 2$.

In addition, a τ-sequence $e_1, \ldots, e_k$ is an *alternating τ-sequence*, if

(S5) no arc among $e_1, \ldots, e_{k-1}$ is incident to C_k', and
(S6) $\mathsf{deg}_{\tau_{k-1}}(C_k') \leq 2$,

and an alternating τ-sequence $e_1, \ldots, e_k$ is an *augmenting τ-sequence* if

(S7) $\mathsf{deg}_{\tau_{k-1}}(C_k') = 1$.

This definition immediately implies the following statement.

Observation 8. *If $e_1, \ldots, e_k$ is an augmenting τ-sequence, then for $\tau' = \tau \div e_1 \div e_2 \div \ldots \div e_k$, we have $\#\mathsf{leaves}(\tau) > \#\mathsf{leaves}(\tau')$.* □

Also, we have the following useful observation which we shall need later. (The proof is omitted due to the length restriction.)

Proposition 9. *If a τ-sequence $e_1, \ldots, e_k$ satisfies (S7), then $e_1, \ldots, e_k$ is an augmenting τ-sequence.* □

We now prove the following theorem which is the first of the two ingredients in our polynomial time algorithm for the leafage.

Theorem 10. *Let T and T^* be two clique trees of G such that T has more leaves than T^*. Then there exists an augmenting ε_T-sequence.*

Proof. We define the *distance* from T to T^* to be the value

$$\mathsf{dist}(T, T^*) = \sum_{CC' \in E(T)} \Big(d_{T^*}(C, C') - 1 \Big)$$

where $d_{T^*}(C, C')$ denotes the distance between C and C' in T^*.

The proof is by induction on $\mathsf{dist}(T, T^*)$. Since T has more leaves than T^* and both have the same number of edges, there must exist C such that the degree of C in T is at least three and is strictly larger than the degree of C in T^*. This implies that if $T_1, \ldots, T_k$ are the components of $T - C$,

there exists i such that no vertex of T_i is adjacent to C in T^*. Let C' be the vertex of T_i that is adjacent to C in T. Let $\mathcal{A}$ and $\mathcal{B}$ be the vertices of the two connected components we obtain by removing the edge CC' from T; we can assume $C \in \mathcal{A}$ and $C' \in \mathcal{B}$. Let P be the path from C to C' in T^*. Since $C \in \mathcal{A}$ and $C' \in \mathcal{B}$, there exists an edge C^*C^{**} of P such that $C^* \in \mathcal{A}$ and $C^{**} \in \mathcal{B}$. By the definition of T_i, we have $C^* \neq C$. Let $S = C \cap C'$ and $S^* = C^* \cap C^{**}$. Since C^*C^{**} is on the path from C to C' in T^*, we have $S \subseteq S^*$. Also, CC' is on the path from C^* to C^{**} in T, since $C^* \in \mathcal{A}$ and $C^{**} \in \mathcal{B}$. Hence, $S^* \subseteq S$ and consequently $S = S^*$. We observe that $C^* \cap C' \subseteq C^* \cap C^{**} = S$, since the edge C^*C^{**} lies on the path from C^* to C'. Hence, $C^* \cap C' = S$, since $C^* \supseteq S^* = S$ and $C' \supseteq S$. Finally, by Theorem 3, we have that C^*C' is an edge of $\mathcal{C}_r(G)$.

Next, let T' denote the tree we obtain by removing the edge CC' from T and adding the edge C^*C'. By Theorem 1, T' is a clique tree of G, since T is. This implies that $e = CC^*$ is an arc of $\mathcal{D}_{\varepsilon_T}$ with label S, since $\varepsilon_{T'} = \varepsilon_T \div e$. Recall that $\mathsf{deg}_{\varepsilon_T}(C) \geq 3$ by the choice of C. If in addition $\mathsf{deg}_{\varepsilon_T}(C^*) = 1$, then e is an augmenting ε_T-sequence, and we are done. Therefore, we can assume that $\mathsf{deg}_{\varepsilon_T}(C^*) \geq 2$. We now observe that $\mathsf{dist}(T', T^*) < \mathsf{dist}(T, T^*)$, since $\mathsf{dist}(T', T^*) - \mathsf{dist}(T, T^*) = d_{T^*}(C^*, C') - d_{T^*}(C, C') < 0$ because $C^* \neq C$ and C^* is on the path from C to C' in T^*. Hence, by induction, there exists an augmenting $\varepsilon_{T'}$-sequence $e_1, \ldots, e_k$ which implies that $e, e_1, \ldots, e_k$ is an augmenting ε_T-sequence. □

4.3 Paths

A directed path $C_1, C_2, \ldots, C_k$ in $\mathcal{D}_\tau$ is an *alternating path* of $\mathcal{D}_\tau$, if

(i) $\mathsf{deg}_\tau(C_1) \geq 3$, and
(ii) $\mathsf{deg}_\tau(C_j) \geq 2$ for each $2 \leq j \leq k-1$.

An alternating path $C_1, C_2, \ldots, C_k$ of $\mathcal{D}_\tau$ is an *augmenting path* of $\mathcal{D}_\tau$, if

(iii) $\mathsf{deg}_\tau(C_k) = 1$.

In what follows, we present the second of the two main ingredients we need for our algorithm for the leafage. In particular, we show that whenever an augmenting τ-sequence exists (for instance, by Theorem 10), we can find a directed path in $\mathcal{D}_\tau$ starting from a vertex of degree at least three to a leaf of τ, i.e., an augmenting path, and conversely, whenever such a path P in $\mathcal{D}_\tau$ exists, we can use it to obtain a τ-sequence starting and ending at the same vertices as P, i.e., an augmenting τ-sequence. In both cases, we construct the sequence (path) by incrementally adding edges one by one; we show that whenever we get stuck, there will be a

"shortcut" in the sequence (path) which will allow us to continue this process until a desired path (sequence) is obtained.

Finally, we note that this property (combined with Theorem 10) reduces the problem of leafage to the problem of finding a directed path in a digraph which is what allows us to solve the problem in polynomial time (for the details of the algorithm, see the next section).

Theorem 11. *There exists an augmenting path in $\mathcal{D}_\tau$ if and only if there exists an augmenting τ-sequence.*

Proof. For the forward direction, let $P = C_1, C_2, \ldots, C_k$ be an augmenting path of $\mathcal{D}_\tau$. Let $\mathcal{F}$ denote the subgraph of $\mathcal{D}_\tau$ induced on $C_1, \ldots, C_k$, and let P' be a shortest directed path in $\mathcal{F}$ from C_1 to C_k. It is easy to verify that P' is also an augmenting path of $\mathcal{D}_\tau$. (Possibly $P' = P$.)

Let $e_1, \ldots, e_{k'}$ be the arcs of P' in the order they appear on P'. That is, e_1 is incident to C_1, and for each $1 \leq i < k'$, the arcs e_i and e_{i+1} share an end-point. Now, using Proposition 7 and the minimality of P', it follows that $e_1, \ldots, e_{k'}$ is an augmenting τ-sequence (details omitted).

For the backward direction, we need the following stronger claim $(*)$.

If $e_1, \ldots, e_k$ is a alternating τ-sequence, then there exists an alternating path of $\mathcal{D}_\tau$ that ends in C'_k where $e_k = C_k C'_k$ such that each vertex of this path is incident to some arc among $e_1, \ldots, e_k$. $(*)$

This claim can be proved by induction on k. (We omit further details of this proof due to the length restriction.)

Finally, let $e_1, \ldots, e_k$ be an augmenting τ-sequence where $e_1 = C_1 C'_1$, $\ldots$, $e_k = C_k C'_k$, and $\tau_0, \ldots, \tau_k$ are token mappings such that $\tau_0 = \tau$ and $\tau_i = \tau_{i-1} \div e_i$ for each $1 \leq i \leq k$. From (S5-S7), we have $\mathsf{deg}_{\tau_{k-1}}(C'_k) = 1$ and no arc among $e_1, \ldots, e_{k-1}$ is incident to C'_k. This yields $\mathsf{deg}_\tau(C'_k) = \mathsf{deg}_{\tau_0}(C'_k) = 1$. Now, by $(*)$, there exists an alternating path P of $\mathcal{D}_\tau$ that ends in C'_k, and since $\mathsf{deg}_\tau(C'_k) = 1$, the path P is also an augmenting path of $\mathcal{D}_\tau$. That concludes the proof. □

5 Algorithm

Now, we are finally ready to prove the main theorem of this paper.

Theorem 12. *There exists an $O(n^3)$ time algorithm that, given a chordal graph G, computes $l(G)$ and a tree model of G with $l(G)$ leaves.*

Proof. The algorithm goes as follows.

(1) Start by computing some clique tree T of G. Then construct the extended degree mapping ε_T of T, and let $\tau = \varepsilon_T$.
(2) Construct the digraph $\mathcal{D}_\tau$.
(3) Search in $\mathcal{D}_\tau$ for a shortest directed path P from a vertex of degree at least three in τ to a leaf of τ.
(4) If such path P is found, then consider the arcs of P one by one, and for each arc $e = CC'$ of P, if S is the label of e, remove a token with label S from $\tau(C)$ and add it to $\tau(C')$. Then go back to step (2).
(5) If such directed path P is not found, then construct a clique tree T corresponding to τ, that is, a tree T with $\tau = \varepsilon_T$. Then output T and the number of leaves of T.

The correctness of this algorithm follows from Theorems 10, and 11, and the observation that the path P in step (4) is neccessarily an augmenting path of $\mathcal{D}_\tau$, and therefore, Observation 8 implies that the new mapping τ has less leaves. We now discuss the complexity.

Recall that G contains at most n maximal cliques. By [6], we can construct a clique tree T of G and the mapping ε_T in time $O(n^2)$. To simplify processing during the algorithm, we precompute the connected components of $\mathcal{H}_S$ for each minimal separator S. This can be accomplished directly in time $O(n^3)$, since G has $O(n)$ minimal separators.

Next, we observe that there are precisely $2n-2$ tokens of τ and at most $n-1$ ways to move each of them. Hence, $\mathcal{D}_\tau$ contains $O(n^2)$ arcs. In fact, for a token with label S belonging to a clique C, we can find all arcs with label S going out of C in $\mathcal{D}_\tau$ by exploring $\mathcal{H}_S$ and testing conditions of Proposition 7. This can be easily accomplished in time $O(n)$ using the precomputed connected components of $\mathcal{H}_S$. Altogether, constructing $\mathcal{D}_\tau$ takes $O(n^2)$ time. The next step, finding the path P in $\mathcal{D}_\tau$, can be accomplished using a breadth-first search of $\mathcal{D}_\tau$ in time $O(n^2)$. If P is found, we can construct the new mapping τ directly in time $O(n)$.

We repeat the above steps at most n times since T has at most n leaves, and therefore, $O(n^3)$ time altogether.

Finally, if P is not found, we construct a tree T with $\varepsilon_T = \tau$ using the proof of Theorem 6 in time $O(n^2)$. This follows from the fact that G has $O(n)$ minimal separators and from an observation that constructing a tree from a degree mapping (see Theorem 5) takes $O(n)$ time. □

6 Conclusions

We have presented the first polynomial time algorithm for the problem of computing the leafage of a chordal graph. We showed that the algorithm

runs in time $O(n^3)$, however, our complexity analysis was not very tight. Therefore, it seems likely that a more efficient, perhaps $O(n^2)$ or linear time implementation can be found. Furthermore, the algorithm provides no certificate for the minimality of the output. We believe that because of the min-max character of the problem it should be possible to characterize the dual of the problem which in turn can be used for certification. Lastly, we remark that our algorithm shows that computing a minimum leaf maximum weight spanning tree is polynomial time solvable for the class of reduced clique graphs whereas this problem is NP-hard in general [11] because the case of at most two leaves is the Hamiltonian path problem.

References

1. Booth, K.S., Lueker, G.S.: Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. J. Comput. Syst. Sci. **13** (1976) 335–379
2. Broersma, H., Fomin, F.V., Nešetřil, J., Woeginger, G.J.: More about subcolorings. Computing **69** (2002) 187–203
3. Fulkerson, D.R., Gross, O.A.: Incidence matrices and interval graphs. Pacific Journal of Mathematics **15** (1965) 835–855
4. Galinier, P., Habib, M., Paul, C.: Chordal graphs and their clique graphs. In: Graph-Theoretic Concepts in Computer Science (WG'95), Lecture Notes in Computer Science 1017, Springer-Verlag (1995) 358–371
5. Gavril, F.: The intersection graphs of subtrees in trees are exactly the chordal graphs. Journal of Combinatorial Theory B **16** (1974) 47–56
6. Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York (1980)
7. Habib, M., Stacho, J.: Linear algorithms for chordal graphs of bounded directed vertex leafage. In: DIMAP Workshop on Algorithmic Graph Theory, Electronic Notes in Discrete Mathematics **32** (2009) 99–108
8. Kloks, T., Kratsch, D., Müller, H.: Asteroidal sets in graphs. In: Graph-Theoretic Concepts in Computer Science (WG'97), Lecture Notes in Computer Science 1335, Springer Berlin/Heidelberg (1997) 229–241
9. Lin, I.J., McKee, T.A., West, D.B.: The leafage of a chordal graph. Discussiones Mathematicae Graph Theory **18** (1998) 23–48
10. Prisner, E.: Representing triangulated graphs in stars. Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg **62** (1992) 29–41
11. Rahman, M.S., Kaykobad, M.: Complexities of some interesting problems on spanning trees. Information Processing Letters **94** (2005) 93–97
12. Stacho, J.: On 2-subcolourings of chordal graphs. In: LATIN 2008: Theoretical Informatics, Lecture Notes in Computer Science 4957. (2008) 520–530
13. Stacho, J.: Complexity of subcolourings of chordal graphs. (2009) manuscript.
14. West, D.B. personal communication (2008)
15. West, D.B.: Introduction to Graph Theory (Second edition). Prentice Hall (2001)