A CLIQUE TREE ALGORITHM FOR PARTITIONING A CHORDAL GRAPH INTO TRANSITIVE SUBGRAPHS *

Dedicated to Miroslav Fiedler and Vlastimil Pták on the occasion of their retirement.

BARRY W. PEYTON†, ALEX POTHEN‡ AND XIAOQING YUAN§

Abstract.

A partitioning problem on chordal graphs that arises in the solution of sparse triangular systems of equations on parallel computers is considered. Roughly the problem is to partition a chordal graph G into the fewest transitively orientable subgraphs over all perfect elimination orderings of G, subject to a certain precedence relationship on its vertices. In earlier work, a greedy scheme that solved the problem by eliminating a largest subset of vertices at each step was described, and an algorithm implementing the scheme in time and space linear in the number of edges of the graph was provided. A more efficient greedy scheme, obtained by representing the chordal graph in terms of its maximal cliques, is described here. The new greedy scheme eliminates in a specified order a largest set of "persistent leaves", a subset of the leaf cliques in the current graph, at each step. Several new results about minimal vertex separators in chordal graphs, and in particular the concept of a *critical separator* of a leaf clique, are employed to prove that the new scheme solves the partitioning problem. We provide an algorithm implementing the scheme in time and space linear in the size of the clique tree.

AMS(MOS) subject classifications: primary 65F50, 65F05, 68R10.

Keywords. chordal graph, clique tree, critical separator, directed acyclic graph, perfect elimination ordering, sparse triangular solution, vertex separator, transitive closure, transitive perfect elimination ordering.

---

* Submitted to *Linear Algebra and its Applications*. Written July 1993; revised Oct. 1994.

† Mathematical Sciences Section, Oak Ridge National Laboratory, P. O. Box 2008, Bldg. 6012, Oak Ridge, TN 37831-6367 (`peyton@msr.epm.ornl.gov`). The work of this author was supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research, U. S. Department of Energy under Contract No. DE-AC05-85OR21400.

‡ Department of Computer Science, Old Dominion University, Norfolk, VA 23529-0162 and ICASE, NASA Langley Research Center, Hampton, VA 23681-0001 (`pothen@cs.odu.edu`, `pothen@icase.edu`). This author was supported by U. S. National Science Foundation grant CCR-9024954, by U. S. Department of Energy grants DE-FG02-91ER25095 and DE-FG05-94ER25216, and by the Canadian Natural Sciences and Engineering Research Council under grant OGP0008111 at the University of Waterloo.

§ IBM Canada Lab, 1150 Eglinton Ave. East, North York, Ontario, M3C 1H7 Canada.

1. Introduction. We consider a partitioning problem on chordal graphs that arises in the design of parallel algorithms for solving sparse triangular systems of equations. Given a chordal graph G with its vertices numbered in a perfect elimination ordering (PEO), we obtain a directed acyclic graph (DAG) by directing every edge from its lower-numbered to its higher-numbered endpoint. (Definitions of chordal graph terms are included in the next section.) Roughly the problem is to partition the chordal graph G into the fewest transitively closed subgraphs, subject to a certain precedence relationship on the vertices, over all $DAGs$ that may be obtained from $PEOs$ of G in this manner. In earlier work [18] we designed a greedy algorithm for solving this problem that uses an adjacency list representation of the graph. Here we describe another, more efficient, greedy algorithm obtained by viewing the chordal graph as a collection of maximal cliques.

We will need to introduce some notation before we can state the problem more precisely.

Let $G_d = (V, F)$ be a DAG. If there exists a directed path from a vertex j to another vertex i in G_d, then j is a *predecessor* of i, and i is a *successor* of j. Given a set $X \subseteq V$, let $F_X \subseteq F$ be the set comprising every edge directed from a vertex in X to any vertex in the graph. The *edge subgraph* induced by F_X is the subgraph of G_d with edge set F_X and vertex set consisting of all vertices which are end-points of these edges. (We will call this the edge subgraph induced by X.) A directed graph is *transitively closed* or *transitive* if the existence of edges (u, v) and (v, w) implies the existence of edge (u, w).

The chordal graph partitioning problem is the following:

PROBLEM 1. *Given a chordal graph $G = (V, E)$, compute a PEO, the associated DAG G_d, and an ordered partition $R_1, R_2, \cdots, R_t$ of its vertices such that*

1. *for every $v \in V$, if $v \in R_i$ then all predecessors of v belong to $R_1, \ldots, R_i$;*
2. *the edge subgraph induced by each R_i is transitively closed; and*
3. *t is minimum over partitions of all $DAGs$ obtained from $PEOs$ of G.*

Problem 1 and a simpler DAG partitioning problem arose in the design of algorithms for solving sparse triangular systems of equations on highly parallel computers. The papers [2, 12, 18, 20] discuss various aspects of this problem, and a survey is provided in [1].

An algorithm for solving this partitioning problem in time and space $\mathcal{O}(|V| + |E|)$ has been described in [18]. This greedy algorithm eliminates all vertices that are 'eligible' for elimination at each step; hence the set of vertices eliminated at the ith step, R_i, has the largest cardinality possible. Let $G_i = G \setminus \cup_{j=1}^{i-1} R_j$ denote the reduced graph at the beginning of the ith step. The set R_i includes all the simplicial vertices of G_i; in addition, it includes the *neosimplicial vertices* of G_i, a subset of the vertices that become newly simplicial when the simplicial vertices of G_i are eliminated. (A precise definition will be given in Section 2.)

Here we present a more efficient greedy algorithm that can be implemented using a clique tree representation of G in $\mathcal{O}(|V| + q)$ time, where $q := \sum_{K \in \mathcal{K}_G} |K|$, and $\boldsymbol{\mathcal{K}}_G$ is the set of maximal cliques of G. The number q is the size of the clique tree, and typically $q \ll |E|$. Since the algorithm is conceptually quite simple, we now provide a high-level description of the algorithm (assuming some knowledge of the clique graph representation of chordal graphs described in Section 3).

Let G_i denote the reduced graph at the beginning of the ith step. The algorithm considers only the leaf cliques in the clique graph representation of G_i for elimination at this step. These cliques are processed by decreasing size of the unique maximal separator contained in each leaf. When a leaf clique K is considered for elimination, all the currently simplicial vertices in K are eliminated in the order in which they became simplicial. It then becomes a non-maximal clique and is deleted from the clique graph. The deletion of K could influence a clique P that contains the maximal separator of K in three ways: If P is a current leaf, and continues to be a leaf after the deletion of K, then the maximal separator size of P is updated as necessary. If P changes from a leaf to a non-leaf, then it is removed from the set of "persistent leaves", and will not be considered for elimination at this step. If P becomes a new leaf, then it will be a candidate for elimination only at the next step. This process of eliminating persistent leaves from the current graph is repeated until the graph is empty.

This "persistent leaf elimination scheme" is a natural greedy algorithm from the clique graph viewpoint in that it deletes all eligible cliques from the current graph at each step. The hard part of the paper is proving that this simple leaf elimination algorithm solves Problem 1. We do this by making a careful study of minimal vertex separators in terms of the clique graph, by introducing the concept of a critical separator, and by partitioning leaves into cohorts using their critical separators.

The rest of this paper is organized into three major parts. The first part, consisting of Sections 2, 3, and 4, develops the fundamental results necessary to characterize the unique first member of maximum cardinality in a vertex partition, R_1. This characterization is obtained in terms of the cliques of G and the minimal vertex separators in G. The second part, which includes Sections 5, 6, and 7, progressively develops a persistent leaf clique elimination scheme that eliminates a subset of vertices that belong to R_1, ordering them in an appropriate ordering. The third part, consisting of Sections 8 and 9, describes a greedy leaf clique elimination algorithm that solves Problem 1 by recursively eliminating persistent leaves at each step. The final section contains a discussion of graphs for which Problem 1 has the solution $R_1 = V$.

We now describe the individual sections in more detail.

In Section 2 we describe the concepts and results from [18] that we require. Section 3 introduces properties of clique intersection graphs, clique trees, and minimal vertex separators of chordal graphs. A vertex v eligible to belong to R_1 was characterized in [18] in terms of the length of a longest chordless path in G in which v is an interior vertex. Section 3 characterizes such a vertex v in terms of the minimal vertex separators of G. The important concept of a critical separator is introduced in Section 4, and a nonsimplicial vertex belonging to R_1 is characterized in terms of critical separators.

Section 5 introduces a simple leaf clique elimination scheme and considers how the set of separators, the set of simplicial vertices, and the set of leaf cliques change upon the elimination of a single clique. In Section 6 this simple elimination scheme is refined by carefully ordering the elimination process, and the cliques and vertices eliminated by the scheme are characterized. It is shown in Section 7 that the refined elimination scheme removes a transitively closed edge subgraph.

In Section 8 we describe a greedy leaf clique elimination scheme that employs the persistent leaf elimination scheme recursively to obtain a solution to Problem 1. An implementation of this greedy scheme that makes uses of a rooted clique tree and runs in $\mathcal{O}(|V| + q)$ time is then briefly described in Section 9.

2. Background. In this section we briefly review chordal graph terminology and the results from [18] that we require in this paper. This section begins the first part of this paper, which includes the next two sections as well. The characterization of the first member of a partition R_1 leads to the concept of a neosimplicial vertex. We characterize neosimplicial vertices in terms of the separators in the chordal graph in the latter sections.

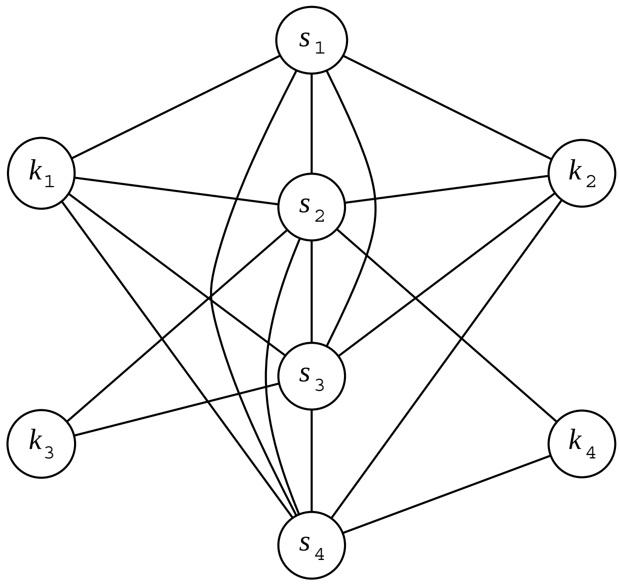
We will assume throughout that the graphs we consider are connected. A *chord* of a cycle (path) in a graph G is an edge of G joining two vertices that are not consecutive on the cycle (path). A graph G is *chordal* if every cycle containing more than three edges has a chord. A cycle or path is *chordless* if it has no chord. Discussions of chordal graphs may be found in Berge [3], Duchet [8], and Golumbic [11]. Peyton [17] and Lundquist [16] discuss the clique graph representation of chordal graphs, and Blair and Peyton [6] provide a recent primer with applications to sparse matrix computations.

An important concept in the solution of Problem 1 is the length of a vertex defined in terms of chordless paths. A vertex v is an *interior* vertex of a path if it lies on the path but is not an endpoint of the path. Any vertex v is either an interior vertex of some chordless path in the graph, or else it is an endpoint of every chordless path on which it lies. In the former case, let $\lambda(v)$ denote the length of a longest chordless path in G which includes v in its interior; note that $\lambda(v) \geq 2$ for all such vertices. In the latter case, define $\lambda(v) = 1$. (We will see later that the latter vertices are *simplicial*, i.e., vertices whose adjacency set is a clique.) We will refer to $\lambda(v)$ as the *length* of a vertex v, and write $\lambda_G(v)$ when we want to make clear that the underlying graph is G.

We will use the chordal graph shown in Figure 2.1 to illustrate various concepts throughout the paper. A "hypergraph" representation of the graph in terms of its maximal cliques is also shown. We use this example throughout this paper to illustrate several new concepts. The hypergraph representation helps to provide insight into the concepts involving separators and leaf cliques. The reader can easily verify from the chordal graph that $\lambda(k_i) = 1$ for $i = 1, \ldots, 4$; $\lambda(s_1) = \lambda(s_2) = 2$; and $\lambda(s_3) = \lambda(s_4) = 3$.

The vertices $v \in V$ for which $\lambda(v) \leq 2$ have certain properties that will play a crucial role in our solution to Problem 1. The first of these is that for such vertices in a chordal graph, there is an interesting partition of $adj(v)$, the adjacency set of v.

The *neighborhood* of a vertex v is $nbd(v) = \{v\} \cup adj(v)$. A vertex $u \in adj(v)$ is said to be *indistinguishable from* v if $nbd(u) = nbd(v)$; the set of neighbors indistinguishable from v will be denoted by $adj^{0}(v)$. A vertex $u \in adj(v)$ is said to *strictly outmatch* v if $nbd(u) \subset nbd(v)$. The set of vertices that strictly outmatch v will be written $adj^{-}(v)$; the set of vertices strictly outmatched by v will be written $adj^{+}(v)$. Finally, let $adj^{*}(v)$



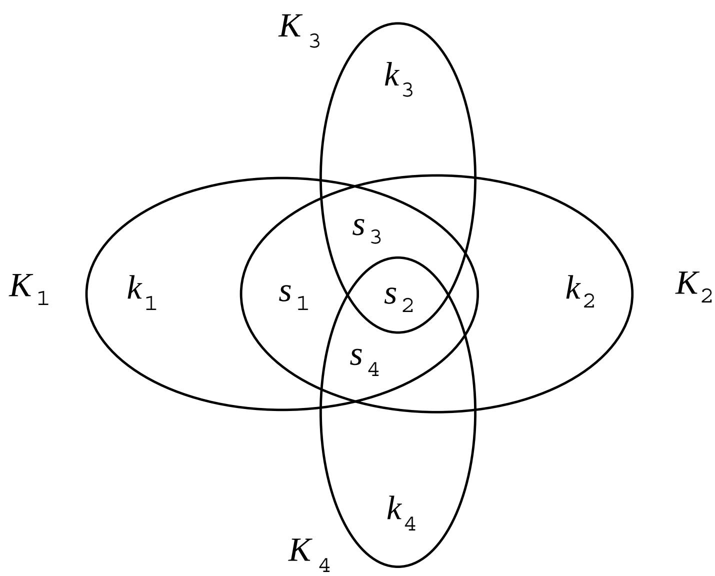


FIG. 2.1. *A chordal graph with maximal cliques* $K_1 = \{k_1, s_1, s_2, s_3, s_4\}$; $K_2 = \{k_2, s_1, s_2, s_3, s_4\}$; $K_3 = \{k_3, s_2, s_3\}$; *and* $K_4 = \{k_4, s_2, s_4\}$. *A hypergraph representation of the graph in terms of its maximal cliques is also shown, since it helps in visualizing the new concepts involving the structure and classification of separators.*

consist of the vertices $u \in adj(v)$ for which $nbd(u)$ and $nbd(v)$ are incomparable. It should be clear that the subsets $adj^-(v)$, $adj^0(v)$, $adj^+(v)$, and $adj^*(v)$ partition $adj(v)$, where v is a vertex in *any* graph G.

LEMMA 2.1. *If v is a vertex of a chordal graph G, then the subsets $adj^-(v)$, $adj^0(v)$, and $adj^+(v)$ partition $adj(v)$ if and only if $\lambda(v) \leq 2$.* □

The second result concerns vertices with length one or two.

LEMMA 2.2. *Let v be a vertex of a chordal graph.*

1. *$\lambda(v) = 1$ if and only if v is simplicial; in which case $adj^-(v) = \emptyset$.*
2. *If $\lambda(v) = 2$ then $|adj^-(v)| \geq 2$, and for every vertex $u \in adj^-(v)$ there exists a vertex $u' \in adj^-(v)$ for which $(u, u') \notin E$.* □

The first, but not the second, of these properties is true for a vertex in any graph. The cycle on four vertices provides a non-chordal counter-example for the latter, since every vertex has $\lambda(v) = 2$ and $adj^-(v)$ equal to the empty set.

We now turn to a characterization of the largest set of vertices whose edge subgraph is transitively closed in the graph G. We need two additional concepts to state the results: *transitive perfect elimination orderings* and *T-sets*.

Let $|V| \equiv n$. An *incomplete ordering of* G relative to a vertex set $X \subseteq V$ is a mapping

$$\alpha : V \rightarrow \{1, 2, \ldots, |X| - 1, |X|, n + 1\}$$

such that α restricted to X is a bijection from X to $\{1, 2, \ldots, |X|\}$ and $\alpha(v) = n + 1$ for each vertex $v \in V - X$. For convenience we shall refer to an incomplete ordering of G as an *ordering of* $G(X)$. (If $X = V$, then we obtain an ordering of the vertices of G.) Given an ordering of the vertices of a graph, we denote by $hadj(v)$ the set of higher-numbered neighbors of v. A *perfect elimination ordering of* $G(X)$ is an ordering of $G(X)$ such that $hadj(v)$ is complete in G for every vertex $v \in X$. (The reader should not confuse $G(X)$ with the subgraph induced by the vertex set X.)

A *transitive ordering* of $G(X)$ is a vertex ordering for which the following property holds: If $\alpha(u) < \alpha(v) < \alpha(w)$ and $(u, v), (v, w) \in E$, then $(u, w) \in E$. Note that the vertices u and v are necessarily taken from X (because $\alpha(u) < \alpha(v) < n + 1$), while the vertex w may be taken from either X or $V - X$.

A *transitive perfect elimination ordering* (TEO) of $G(X)$ is an ordering of $G(X)$ that is both a PEO and a transitive ordering of $G(X)$. Any vertex set $X \subseteq V$ for which there exists a TEO of $G(X)$ shall henceforth be called a *T-set* of G. An example of a T-set is $X = Sim_G \neq \emptyset$, where Sim_G is the set of simplicial vertices of G. It is easy to verify for this example that any ordering of $G(X)$ is a TEO of $G(X)$. The set $X = \{k_1, k_2, k_3, k_4, s_1\}$ is a T-set of the graph in Figure 2.1, since if the vertices are numbered in increasing order as listed, then the ordering is a TEO. Note that X includes a nonsimplicial vertex s_1 of G.

If X is a T-set of G, order the vertices of $G(X)$ in a TEO, and direct each edge that has at least one endpoint in X from the lower to the higher-numbered endpoint. Let E_X denote the subset of edges of G with at least one endpoint in X. Then the edge subgraph of $G(X)$ induced by E_X is a transitively closed subgraph. The following theorem characterizes the largest possible transitively closed subgraph of G that can be obtained in this manner.

THEOREM 2.3. *The unique T-set of maximum cardinality in the graph G is*

$$R = \{v \in V \mid \lambda(v) \leq 2, \text{ and } \lambda(u) \leq 2 \text{ for every } u \in adj^{-}(v)\}. \quad \square \tag{2.1}$$

In the example in Figure 2.1, $R = \{k_1, \ldots, k_4, s_1\}$. The T-set R includes *simplicial vertices*, which are vertices of length one, and *neosimplicial vertices*, vertices v with length two such that vertices that strictly outmatch v have length less than or equal to two as well.

The next result characterizes a greedy solution to Problem 1. Consider reducing the graph G by choosing a T-set $\hat{R}$ of G and removing the vertices in $\hat{R}$ from G in the order specified by a TEO of $G(\hat{R})$; we can then complete the reduction of G to the null graph by applying this process recursively to the reduced graph $G \setminus \hat{R}$.

Suppose the graph G is reduced to the null graph after the removal of t distinct T-sets, each ordered by a TEO. Define $G_1 := G$, and let $G_2, G_3, \ldots, G_{t+1} = \emptyset$ be the sequence of reduced graphs obtained at the end of each "block" elimination step. Let $\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_t$ be the corresponding sequence of T-sets, so that $\hat{R}_i$ is removed from G_i by a TEO of $G_i(\hat{R}_i)$ to obtain the reduced graph $G_{i+1} = G_i \setminus \hat{R}_i$. We shall refer to any partition $\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_t$ obtained by this process as a *T-partition* of V. A PEO α of V can be obtained through this process by ordering for each $1 \leq i \leq t-1$, the vertices in $\hat{R}_{i+1}$ in a TEO after $\hat{R}_i$ has been ordered in a TEO. The resulting PEO is a *compound* TEO of G with respect to the T-partition $\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_t$.

Denote by the *greedy vertex elimination scheme* a scheme that eliminates the maximum cardinality T-set R_i from each graph G_i in this sequence.

THEOREM 2.4. *The greedy vertex elimination scheme generates a minimum-cardinality T-partition of V.* $\square$

3. Clique graphs and vertex lengths.

We begin this section with a description of clique graph and clique tree representations of a chordal graph, and then describe the relationships between vertex separators and clique trees. These will enable us to obtain a result relating vertices with specified lengths to the structure of the separators they belong to.

3.1. Clique trees and separators.

Let the set of maximal cliques of the chordal graph $G = (V, E)$ be denoted by $\mathcal{K}_G$. We define a *clique intersection graph* with vertex set $\mathcal{K}_G$ by joining two cliques K and K' by an edge (K, K') if the intersection $K \cap K'$ is not empty. The *weight* of the edge is the size of the intersection. A *clique tree* $T = (\mathcal{K}_G, \mathcal{E})$ is a maximum weight spanning tree (mst) of the clique intersection graph (Bernstein and Goodman [4]). Every clique tree T of G satisfies the *intersection property*: For every pair of cliques K_1, K_2, the intersection $K_1 \cap K_2$ is contained in every clique on the path joining K_1 and K_2 in T. We denote the set of all clique trees of G by $\mathcal{T}_G$. Background material on clique trees may be found in Blair and Peyton [6].

The maximal cliques of the graph G in Figure 2.1 are listed in the figure caption. The clique intersection graph of G is a complete graph with weight four on edge (K_1, K_2), one on edge (K_3, K_4), and two on all other edges. The clique trees of G are obtained by choosing the edge (K_1, K_2), one edge from the set $\{(K_1, K_3), (K_2, K_3)\}$ and another edge from $\{(K_1, K_4), (K_2, K_4)\}$. Note that the edge (K_3, K_4) belongs to none of the clique trees of G.

Let $\boldsymbol{\mathcal{K}}(u)$ denote the set of maximal cliques of G that contain the vertex u. The following lemma characterizes the adjacency set partition in Lemma 2.1 in terms of the maximal cliques of G.

LEMMA 3.1. *For any pair of vertices u, v of a chordal graph G:*

1. $\boldsymbol{\mathcal{K}}(u) \subset \boldsymbol{\mathcal{K}}(v)$ *if and only if* $u \in adj^{-}(v)$.
2. $\boldsymbol{\mathcal{K}}(u) = \boldsymbol{\mathcal{K}}(v)$ *if and only if* $u \in adj^{0}(v)$.

We omit the simple proof. □

If a and b are non-adjacent vertices in a connected graph G, an *a, b-separator* is a set of vertices S such that a and b belong to two distinct connected components in $G \setminus S$. The set S is a *minimal a, b-separator* if no proper subset of S has this property. We will call S a *minimal vertex separator* or *separator* if it is a minimal a, b-separator for some pair of non-adjacent vertices $a, b \in V - S$.

Ho and Lee ([13], Lemma 2.1) proved the following result.

PROPOSITION 3.2. *The set $S \subset V$ is a minimal vertex separator in the chordal graph G if and only if in every clique tree $T \in \boldsymbol{\mathcal{T}}_G$ there exists some edge (K, K') such that $S = K \cap K'$.* □

The edge (K, K') in the proposition may depend on T. Let $\boldsymbol{\mathcal{K}}_1$ and $\boldsymbol{\mathcal{K}}_2$ denote the sets of cliques in the two subtrees obtained when the edge (K, K') is removed from T. Define $V_1 \subset V$ ($V_2 \subset V$) to be the set of vertices belonging to the cliques in $\boldsymbol{\mathcal{K}}_1$ ($\boldsymbol{\mathcal{K}}_2$) but excluding vertices in S. Then $S = K \cap K'$ is a minimal a, b-separator for any pair $a \in V_1$, $b \in V_2$ (Ho and Lee [13], Lundquist [16]).

The separators in the example are $K_1 \cap K_2 = \{s_1, s_2, s_3, s_4\} \equiv S_1$, $K_1 \cap K_3 = K_2 \cap K_3 = \{s_2, s_3\} \equiv S_2$, and $K_1 \cap K_4 = K_2 \cap K_4 = \{s_2, s_4\} \equiv S_3$. Note that $K_3 \cap K_4 = \{s_2\}$ is *not* a separator since the corresponding edge does not belong to any clique tree of G.

For any clique tree $T = (\boldsymbol{\mathcal{K}}_G, \boldsymbol{\mathcal{E}})$, consider the multiset

$$\boldsymbol{\mathcal{M}}_T = \{K \cap K' \mid (K, K') \in \boldsymbol{\mathcal{E}}\}.$$

From the previous proposition we have that $\boldsymbol{\mathcal{M}}_T$ is a multiset of minimal vertex separators of G. If $T, U \in \boldsymbol{\mathcal{T}}_G$ are two clique trees of G, Ho and Lee further showed that the multisets $\boldsymbol{\mathcal{M}}_T$ and $\boldsymbol{\mathcal{M}}_U$ are identical. Hence we let $\boldsymbol{\mathcal{M}}_G$ denote the multiset of separators associated with every clique tree in $\boldsymbol{\mathcal{T}}_G$.

Let the set of cliques containing a set $S \subseteq V$ be $\boldsymbol{\mathcal{K}}(S) = \{K \in \boldsymbol{\mathcal{K}}_G : S \subseteq K\}$ (usually S will be a separator), and let the set of separators belonging to a clique K be $\boldsymbol{\mathcal{S}}(K) = \{S \in \boldsymbol{\mathcal{M}}_G : S \subset K\}$. The set $\boldsymbol{\mathcal{S}}(K)$ contains one copy of each distinct separator in $\boldsymbol{\mathcal{M}}_G$ contained in K.

In the example in Figure 2.1, each clique K_1 and K_2 contains the separators $S_1 = \{s_1, \ldots, s_4\}$, $S_2 = \{s_2, s_3\}$, and $S_3 = \{s_2, s_4\}$. The set of cliques containing the separator S_2 is $\{K_1, K_2, K_3\}$.

We will require the following lemma in proving subsequent results.

LEMMA 3.3. *If $S \in \boldsymbol{\mathcal{S}}(K)$, then there exists a clique $K' \in \boldsymbol{\mathcal{K}}_G$ such that $S = K \cap K'$; furthermore, S is a minimal u, v-separator for every pair of vertices $u \in K \setminus K'$ and $v \in K' \setminus K$.*

Proof. Let T be a clique tree of the chordal graph G. By Proposition 3.2, since $S \in \boldsymbol{\mathcal{M}}_G$ there exists an edge (K_1, K_2) in the clique tree T such that $S = K_1 \cap K_2$. If either one of these cliques is identical to K, then we are done. Hence assume that K is distinct from these two cliques.

Let $\boldsymbol{\mathcal{K}}_1$ and $\boldsymbol{\mathcal{K}}_2$ denote the sets of cliques in the two subtrees obtained by removing the edge (K_1, K_2) from T, and without loss of generality, let K_1 and K belong to $\boldsymbol{\mathcal{K}}_1$. Since vertices in S belong to both K and K_2, we have that $K \cap K_2 \supseteq S$. From the clique intersection property of clique trees, $K \cap K_2$ is contained in every clique on the path in T from K to K_2, and hence $K \cap K_2$ belongs to K_1. But $K_1 \cap K_2 = S$ implies that $K \cap K_2 = S$. Now the tree T' obtained by replacing the edge (K_1, K_2) by the edge (K, K_2) in T is also a maximum weight spanning tree of the clique intersection graph, and hence is a clique tree. It follows that if we let $K_2 = K'$, then (K, K') is an edge of a clique tree, and thus S is a separator for every pair of vertices $u \in K - K'$ and $v \in K' - K$. □

A clique K is a *leaf clique of* G if there exists a clique tree $T \in \boldsymbol{\mathcal{T}}_G$ in which K is a leaf. Note that such a clique K may not be a leaf in some other clique tree T'. We let $\boldsymbol{\mathcal{L}}_G$ denote the set of leaf cliques of the chordal graph G, and $\boldsymbol{\mathcal{L}}_T$ denote the set of leaves of a specified clique tree T. Blair and Peyton [5] obtained the following characterization of a leaf clique.

PROPOSITION 3.4. *A clique K is a leaf clique of G if and only if it contains a unique separator S that is maximal among the separators in $\boldsymbol{\mathcal{S}}(K)$.* □

We will refer to the unique maximal separator contained in a leaf K as the *leaf separator* of K, and denote it by $Sep_G(K)$. The vertices in a leaf K can be partitioned into two subsets: the set of simplicial vertices which belongs to no other clique, $Sim_G(K)$; and the set of vertices which belongs to other cliques, all contained in the leaf separator $Sep_G(K)$, and therefore contained in some clique K' such that $K \cap K' = Sep_G(K)$. If K is not a leaf of G, then we will call it a *non-leaf*. From Proposition 3.4, a non-leaf clique contains at least two maximal separators that are pairwise incomparable.

In Figure 2.1, $\boldsymbol{\mathcal{L}}_G = \{K_1, \ldots, K_4\}$; $Sim_G(K_i) = \{k_i\}$, for $i = 1, \ldots, 4$; $Sep_G(K_1) = Sep_G(K_2) = S_1 = \{s_1, \ldots, s_4\}$; $Sep_G(K_3) = S_2 = \{s_2, s_3\}$, and $Sep_G(K_4) = S_3 = \{s_2, s_4\}$.

3.2. Vertex lengths and separators. In this subsection, we characterize vertices with specified values of the length parameter, $\lambda(.)$, in terms of the separators in the clique graph. This is one of the central results in this paper. We use $S' \triangle S''$ to denote the set $(S' - S'') \cup (S'' - S')$.

THEOREM 3.5. *For a vertex v in a chordal graph G,*

1. *$\lambda_G(v) = 1$ if and only if v belongs to no separator of G; in this case v is simplicial.*

2. *$\lambda_G(v) = 2$ if and only if v belongs to some separator of G, and $\forall K \in \boldsymbol{\mathcal{K}}_G(v)$, every separator $S' \in \boldsymbol{\mathcal{S}}(K)$ that includes v contains every separator $S'' \in \boldsymbol{\mathcal{S}}(K)$ that does not.*
3. *$\lambda_G(v) \geq 3$ if and only if there exist two incomparable separators $S', S'' \in \boldsymbol{\mathcal{S}}(K)$ in some clique $K \in \boldsymbol{\mathcal{K}}_G(v)$ such that $v \in S' \triangle S''$.*

Proof. Note that the conditions at the left-hand-side of the three items define a tri-partition of V, and likewise, the conditions at the right-hand-side of the three items partition V. The first item is easily proved as follows. From Lemma 2.2, $\lambda(v) = 1$ if and only if v is a simplicial vertex; it is well-known that v is simplicial if and only if it belongs to exactly one maximal clique [14]. Then from Proposition 3.2, v does not belong to any separator. Hence it suffices to prove the third item.

We begin by proving that the right-hand-side implies the left-hand-side in the third item. Suppose that there exists a clique $K \in \boldsymbol{\mathcal{K}}_G(v)$ satisfying the given condition. Without loss of generality let v belong to S', and choose $w \in S'' - S'$. By Lemma 3.3 we can find a clique K' such that $S' = K \cap K'$ separates vertices in $K - S'$ from vertices in $K' - S'$. Similarly we can choose a clique K'' such that $S'' = K \cap K''$ is a separator separating vertices in $K - S''$ from $K'' - S''$. By the maximality of these cliques, choose $k' \in K' - S'$ and $k'' \in K'' - S''$. In the path k', v, w, k'', by the choice of the separators S' and S'', no edge joins w and k' or v and k''. No edge joins k' and k'' since it would create a chordless cycle of length four. Hence the path k', v, w, k'' is chordless, and $\lambda(v) \geq 3$.

We prove the other direction of the third item by contraposition. Negating the condition at the right-hand-side in the third item, either v belongs to no separator in any clique in $\boldsymbol{\mathcal{K}}_G(v)$, or v belongs to at least one separator and in every clique $K \in \boldsymbol{\mathcal{K}}_G(v)$, every separator that includes v contains every separator that does not. The former case has already been considered in the first paragraph of the proof. Suppose now that for every clique $K \in \boldsymbol{\mathcal{K}}_G(v)$, $v \in S' - S''$ implies that $S' \supset S''$ for S', $S'' \in \boldsymbol{\mathcal{S}}(K)$. We will prove that then $\lambda(v) = 2$, thus completing the proof of the theorem.

To obtain a contradiction, suppose $\lambda(v) \geq 3$, and hence that there exists a chordless path u, v, w, x in G. Then u, v belong to some clique K', v, w to another clique K, and w, x to a third clique K''. Further v belongs to every minimal u, w-separator, and w to every minimal v, x-separator in G. Since every separator corresponds to an edge in any clique tree T of G, by Lemma 3.3 we can choose the cliques K, K', and a separator S' such that $S' = K \cap K'$, $v \in S'$, and S' separates $u \in K' - S'$ from $w \in K - S'$. Similarly we choose a clique K'' and a separator S'' such that $S'' = K \cap K''$, $w \in S''$, and S'' separates $v \in K - S''$ from $x \in K'' - S''$.

Together, $v \in S'$ and $v \in K - S''$ imply that $v \in S' - S''$; similarly $w \in K - S'$ and $w \in S''$ imply that $w \in S'' - S'$. But this is a contradiction since we have assumed that $v \in S' - S''$ implies $S' \supset S''$. Since the chordless path of length three containing v as an interior vertex was chosen arbitrarily, this contradiction shows that $\lambda(v) \leq 2$. Since v belongs to some separator, $\lambda(v) = 2$. □

In Figure 2.1, $\lambda(k_i) = 1$, for $i = 1, \ldots, 4$, since these vertices belong to none of the separators; $\lambda(s_1) = 2$, since in both K_1 and K_2, the separator $\{s_1, \ldots, s_4\}$

contains the separators $\{s_2, s_3\}$ and $\{s_2, s_4\}$ that do not include s_1; $\lambda(s_2) = 2$, since s_2 belongs to every separator in G, and hence it satisfies vacuously the second statement in Theorem 3.5; the other vertices have length greater than or equal to three (all of them have length three).

We can now employ Theorem 2.3 to identify the *neosimplicial* vertices of G, i.e., the set of vertices v with length two such that the vertices outmatching v have length less than or equal to two. The set $adj^-(s_1) = \{k_1, k_2\}$; since $\lambda(s_1) = 2$ and $\lambda(k_1) = \lambda(k_2) = 1$, s_1 is a neosimplicial vertex. On the other hand, $adj^-(s_2)$ includes s_3 and s_4, vertices of length three, and hence s_2 is not a neosimplicial vertex.

An easy consequence of the above theorem is the following result.

LEMMA 3.6. *If a nonsimplicial vertex v of a chordal graph G is neosimplicial, then it belongs only to the leaf cliques of G.*

Proof. We prove that if v belongs to a non-leaf clique K of G, then it is not neosimplicial. Since v is not a simplicial vertex, by (2.1) it can belong to R only if $\lambda(v) = 2$; in this case we show that there exists $w \in adj_G^-(v)$ with $\lambda(w) \geq 3$.

The non-leaf clique K contains two maximal incomparable separators S', S''. Choose two vertices $w \in S' - S''$, and $x \in S'' - S'$. Applying Theorem 3.5 to w and x, we find that $\lambda(w)$, $\lambda(x) \geq 3$.

By Theorem 3.5, $\lambda(v) = 2$ implies that every separator belonging to $\boldsymbol{\mathcal{S}}(K)$ that includes v contains every separator that does not. Thus a separator in $\boldsymbol{\mathcal{S}}(K)$ that does not include v is not a maximal separator in $\boldsymbol{\mathcal{S}}(K)$. Hence v belongs to S' and S'', but by the choice of these vertices, $w \notin S''$ and $x \notin S'$. Because $\lambda(v) = 2$, by Lemma 2.1 the sets $adj_G^-(v)$, $adj_G^0(v)$, and $adj_G^+(v)$ partition $adj_G(v)$, and hence w, $x \in adj_G^-(v)$, completing the proof. □

The results in this section imply that an algorithm for eliminating a maximum cardinality T-set need consider only vertices in the leaf cliques.

4. Critical separators. In this section we characterize neosimplicial vertices in terms of the clique graph. More precisely, based on the separators in a leaf we partition nonsimplicial vertices in the leaf into those vertices that are neosimplicial and those that are not. Towards this end, we introduce the concept of a *critical separator*, and partition the leaf cliques into groups called *cohorts* based on their critical separators.

Recall that a leaf clique K contains a unique maximal separator, say S_1, that properly contains every other separator belonging to $\boldsymbol{\mathcal{S}}(K)$. We now order the separators of a leaf clique K as shown:

$$\boldsymbol{\mathcal{S}}(K) = \{S_1 \supset S_2 \supset \cdots \supset S_\ell \supset S_{\ell+1}, S_{\ell+2}, \cdots, S_m\}, \tag{4.1}$$

with the index ℓ chosen as large as possible. (This notation means that $S_\ell \supset S_j$ for every j such that $\ell + 1 \leq j \leq m$.) By the choice of ℓ, for $\ell + 1 \leq j \leq m$, no separator S_j contains every other separator in this set. Choose the largest index $1 \leq r \leq \ell$ such that $\boldsymbol{\mathcal{K}}(S_i) \subseteq \boldsymbol{\mathcal{L}}_G$ for $i = 1, \ldots, r-1$; we define S_r to be the *critical separator* $C(K)$ of the leaf clique K.

If $\ell = 1$ then $r = 1$, and the leaf separator S_1 vacuously satisfies the condition that all lower numbered separators are contained in leaf cliques. Furthermore, if $r = \ell = m$,

S_m also satisfies the definition of the critical separator. Thus a leaf clique always has a critical separator, but this notion is undefined for a non-leaf clique.

The hypergraph representation of the maximal cliques of the chordal graph in Figure 2.1 is particularly helpful in visualizing the following results. Since the separators in K_1 can be ordered as

$$S_1 = \{s_1, \ldots, s_4\} \supset S_2 = \{s_2, s_3\},\ S_3 = \{s_2, s_4\},$$

S_1 is its critical separator. Similarly, S_1 is the critical separator of K_2, S_2 is the critical separator of K_3, and S_3 is the critical separator of K_4.

The importance of critical separators is that they aid in distinguishing between neosimplicial vertices, the nonsimplicial vertices that belong to R, and those that do not. Let a *subcritical separator* of K denote any separator properly contained in the critical separator $C(K) = S_r$; i.e., a separator S_j, where $r + 1 \leq j \leq m$. Further let a *supercritical separator* of K denote the critical separator of K or a separator of K that properly contains the critical separator.

The next theorem characterizes neosimplicial vertices, and is another central result in this paper.

THEOREM 4.1. *A nonsimplicial vertex v of the graph G is neosimplicial if and only if (i). $\boldsymbol{\mathcal{K}}_G(v) \subseteq \boldsymbol{\mathcal{L}}_G$, and (ii). $\forall K \in \boldsymbol{\mathcal{K}}_G(v)$, v belongs only to the supercritical separators of K.*

Proof. First we prove that the left-hand-side implies the right-hand-side by contraposition. If v belongs to a non-leaf, then it cannot be neosimplicial by Lemma 3.6. Hence assume that v belongs only to leaf cliques of G. We proceed to show that if v belongs to a subcritical separator S in a leaf clique K, then it cannot be neosimplicial. The existence of a subcritical separator implies either that the critical separator $C(K)$ is contained in a non-leaf clique, or that $C(K)$ properly contains two incomparable separators S, S' that are maximal among the subcritical separators in K. The former case contradicts our assumption that v does not belong to a non-leaf clique.

Hence consider the latter case. Since v is neosimplicial, by (2.1) we can assume that $\lambda(v) = 2$. Now if $v \in S - S'$, then by Theorem 3.5, $\lambda(v) \geq 3$, hence we must have $v \in S \cap S'$. Choose vertices $s \in S - S'$, $s' \in S' - S$. Applying Theorem 3.5 to s and s', we find that $\lambda(s)$, $\lambda(s') \geq 3$. Since $\lambda(v) = 2$, we have by Lemma 2.1 that $adj^-(v)$, $adj^0(v)$, and $adj^+(v)$ partition $adj(v)$. Now $v \in S \cap S'$ and $s \in S - S'$ imply that $s \in adj^-(v)$. Similarly $s' \in adj^-(v)$. It follows that $v \notin R$ by Theorem 2.3.

To prove the other direction, choose a clique $K \in \boldsymbol{\mathcal{K}}_G(v) \subseteq \boldsymbol{\mathcal{L}}_G$. Order the separators in $\boldsymbol{\mathcal{S}}(K)$ as in (4.1), and let $S_r = C(K)$ denote the critical separator of K. Then $v \in S_1, \ldots, S_q$ where $q \leq r$, since v does not belong to a subcritical separator. The ordering of the separators in (4.1) ensures that there do not exist incomparable separators S', $S'' \in \boldsymbol{\mathcal{S}}(K)$ such that $v \in S' \triangle S''$. Since this is true for every clique $K \in \boldsymbol{\mathcal{K}}_G(v)$, by Theorem 3.5 it follows that $\lambda(v) \leq 2$. Furthermore, since v is nonsimplicial, $\lambda(v) = 2$.

If $u \in adj^-(v)$, then $\boldsymbol{\mathcal{K}}_G(u) \subset \boldsymbol{\mathcal{K}}_G(v)$ by Lemma 3.1, and thus $u \in S_1, \ldots, S_p$ where $p < q$. Repeating the argument given for v in the previous paragraph for the vertex u, we obtain $\lambda(u) \leq 2$. By Theorem 2.3 it follows that v is neosimplicial. $\square$

Thus in the example, s_1 is neosimplicial since it does not belong to the subcritical separators S_2 and S_3 in the cliques K_1 and K_2, while s_2 is not neosimplicial, since it does.

Let $\boldsymbol{\mathcal{C}} = \{C_1, \ldots, C_p\}$ denote the set of critical separators of a chordal graph G. The leaf cliques $\boldsymbol{\mathcal{L}}_G$ can be partitioned into p *cohorts* such that $\boldsymbol{\mathcal{L}}(C)$ includes all the leaves whose critical separator is C. We say that a separator S (a vertex v) belongs to a cohort $\boldsymbol{\mathcal{L}}(C)$ if S (the vertex v) is contained in some clique in the cohort.

In our example, $\boldsymbol{\mathcal{L}}(S_1) = \{K_1, K_2\}$, $\boldsymbol{\mathcal{L}}(S_2) = \{K_3\}$, and $\boldsymbol{\mathcal{L}}(S_3) = \{K_4\}$.

LEMMA 4.2. *Let $\boldsymbol{\mathcal{L}}(C_1)$ and $\boldsymbol{\mathcal{L}}(C_2)$ be two distinct cohorts corresponding to critical separators C_1 and C_2, respectively, such that $C_1 \not\subset C_2$. If S is a supercritical separator contained in some clique in $\boldsymbol{\mathcal{L}}(C_1)$, then S cannot be a supercritical separator of any clique in $\boldsymbol{\mathcal{L}}(C_2)$.*

Proof. First we show that C_1 cannot belong to any clique in $\boldsymbol{\mathcal{L}}(C_2)$. Assume, to obtain a contradiction, that $K \in \boldsymbol{\mathcal{L}}(C_2)$ contains C_1. Then since C_2 is the critical separator of K, either C_1 is a supercritical separator in K, in which case $C_1 \supset C_2$, or C_1 is a subcritical separator in K, and we would have $C_1 \subset C_2$. By assumption the latter relationship cannot be true. If $C_1 \supset C_2$, then we claim that C_1 and not C_2, would be the critical separator of K.

Since $C_1 \supset C_2$, every $K' \in \boldsymbol{\mathcal{L}}(C_1)$ contains C_2 as a subcritical separator. Now C_1 is the critical separator of K' either because there is a non-leaf clique containing C_1, or because there exist two or more maximal separators contained in C_1 that are incomparable. However, then since C_1 belongs to K, the same situation would apply to K, and C_1 would be the critical separator of K.

Now suppose $K' \in \boldsymbol{\mathcal{L}}(C_1)$ contains a supercritical separator $S \supset C_1$. If S were also contained in a clique $K \in \boldsymbol{\mathcal{L}}(C_2)$, then K and K' would have C_1 in common, which we have just proved cannot happen. This completes the proof. □

LEMMA 4.3. *A neosimplicial vertex v belongs to a subset of the cliques in exactly one cohort.*

Proof. Suppose the cliques containing the vertex v, $\boldsymbol{\mathcal{K}}_G(v)$, belong to $q \geq 2$ distinct cohorts $\boldsymbol{\mathcal{L}}(C_1)$, ..., $\boldsymbol{\mathcal{L}}(C_q)$. Choose any clique tree T of G, and let T_v denote the subtree induced by the cliques in $\boldsymbol{\mathcal{K}}_G(v)$, where v is a neosimplicial vertex. The edges of this tree correspond to minimal vertex separators in G. Since T_v is a tree whose vertices are cliques belonging to q different cohorts, there must be an edge in T_v joining a clique in some cohort to a clique in some other cohort. But two cliques in distinct cohorts cannot have a separator that is supercritical in both of them by the previous lemma, and hence any tree edge joining a clique K in one cohort to a clique K' in a second cohort must correspond to a subcritical separator in one of them. Then v belongs to this subcritical separator, and by Theorem 4.1 cannot be neosimplicial. □

The example in Figure 2.1 again provides an illustration. The neosimplicial vertex s_1 belongs only to $\boldsymbol{\mathcal{L}}(S_1)$, while s_2 belongs to three cohorts $\boldsymbol{\mathcal{L}}(S_i)$ for $i = 1$, 2 and 3, and hence is not neosimplicial.

5. A leaf elimination scheme. The following three sections constitute the second part of the paper where we develop a persistent leaf elimination scheme that removes a subset of the maximum cardinality T-set R from the graph G. To aid in understanding, we begin in this section with a description of the simple clique elimination framework used for this purpose, and then refine it in the next section.

$P\text{-}leaves \leftarrow \boldsymbol{\mathcal{L}}_G$;
$H \leftarrow G$;
while $P\text{-}leaves \neq \emptyset$ **do**
    Choose a clique $K \in P\text{-}leaves$;
    Choose a clique P such that $P \cap K = Sep_H(K)$;
    $H \leftarrow H \setminus Sim_H(K)$;
    $P\text{-}leaves \leftarrow P\text{-}leaves - \{K\}$;
    if $P \notin \boldsymbol{\mathcal{L}}_H$ **then** $P\text{-}leaves \leftarrow P\text{-}leaves - \{P\}$;
end while

FIG. 5.1. *A leaf elimination scheme.*

This elimination scheme is shown in Figure 5.1. It considers only the leaf cliques of G as candidates for deletion, and eliminates the simplicial vertices from each leaf clique chosen for removal. The rest of this section considers how various sets of cliques, separators, and vertices change when a leaf clique is eliminated.

Let $K_1, \ldots, K_p$ be the set of leaves eliminated by this scheme, listed in the order of elimination. Let $G \equiv H_0$, and for $j = 1, \ldots, p$, let H_j be the reduced graph obtained by eliminating the simplicial vertices in the leaf K_j from the graph H_{j-1}. We denote the final reduced graph $H_p \equiv G^+$. In the results that follow, we let H denote a graph H_j and H^+ denote H_{j+1}, the next reduced graph in the sequence.

Results similar to the next two lemmas may be found in Blair and Peyton [5] (Section 4.2, Lemma 9; and Section 5.1, Lemma 13), and hence we omit their proofs.

The first result shows how the multiset of separators and the set of cliques change after elimination of *any clique*.

LEMMA 5.1. *Let $K \in \boldsymbol{\mathcal{K}}_H$ be* any *maximal clique in a chordal graph H, and let $H^+ = H \setminus Sim_H(K)$ be the reduced graph obtained by the elimination of simplicial vertices in K. Then*

1. *$\boldsymbol{\mathcal{M}}_{H^+} = \boldsymbol{\mathcal{M}}_H - \{S\}$, and $\boldsymbol{\mathcal{K}}_{H^+} = \boldsymbol{\mathcal{K}}_H - \{K\}$, where $S = K - Sim_H(K)$, if and only if $K \in \boldsymbol{\mathcal{L}}_H$; in this case, $S = Sep_H(K)$, the leaf separator of K.*
2. *$\boldsymbol{\mathcal{M}}_{H^+} = \boldsymbol{\mathcal{M}}_H$, and $\boldsymbol{\mathcal{K}}_{H^+} = \boldsymbol{\mathcal{K}}_H - \{K\} \cup \{K^+\}$, where $K^+ = K - Sim_H(K)$, if and only if $K \notin \boldsymbol{\mathcal{L}}_H$.* □

The next result characterizes changes in the set of simplicial vertices and the set of separators in a clique when a leaf clique is eliminated.

LEMMA 5.2. *Let S be the leaf separator of a leaf clique K in a chordal graph H, and let P be a clique with $S \in \boldsymbol{\mathcal{S}}_H(P)$. Let $H^+ = H \setminus Sim_H(K)$ be the reduced graph obtained by eliminating the simplicial vertices in K from H.*

1. *If $|\boldsymbol{\mathcal{K}}_H(S)| \geq 3$, then $\forall K' \in \boldsymbol{\mathcal{K}}_{H^+}$, we have $Sim_{H^+}(K') = Sim_H(K')$. Further, if S is a maximal separator in $\boldsymbol{\mathcal{M}}_H$, then $\forall K' \in \boldsymbol{\mathcal{K}}_{H^+}$, we have $\boldsymbol{\mathcal{S}}_{H^+}(K') =$*

$\boldsymbol{\mathcal{S}}_H(K')$.

2. If $|\boldsymbol{\mathcal{K}}_H(S)| = 2$, then

(a) $\forall K' \in \boldsymbol{\mathcal{K}}_{H^+} - \{P\}$, we have $\boldsymbol{\mathcal{S}}_{H^+}(K') = \boldsymbol{\mathcal{S}}_H(K')$, and $Sim_{H^+}(K') = Sim_H(K')$, and

(b) $\boldsymbol{\mathcal{S}}_{H^+}(P) = \boldsymbol{\mathcal{S}}_H(P) - S$; $Sim_{H^+}(P) = Sim_H(P) \cup \delta S$, where δS is the subset of vertices in S that belong only to K and P. □

As cliques are deleted in the elimination scheme, a non-leaf clique K' may become a leaf clique in the reduced graph. Such cliques do not contain any neosimplicial vertices by Lemma 3.6, and we will show later that the simplicial vertices in these cliques need not be eliminated at the current step to solve Problem 1. The next result describes when a leaf clique in the current graph can become a non-leaf in the reduced graph.

LEMMA 5.3. *Let $S = Sep_H(K)$ be the leaf separator of a leaf clique K in a chordal graph H, and let P be another leaf clique of H such that $S \in \boldsymbol{\mathcal{S}}_H(P)$. Then P is a non-leaf clique in the reduced graph $H^+ = H \setminus Sim_H(K)$ if and only if (i) $\boldsymbol{\mathcal{K}}_H(S) = \{K, P\}$, (ii) S is the critical separator of P, and (iii) P contains subcritical separators.*

Proof. When $Sim_H(K)$ is eliminated K is deleted as a maximal clique, and since $\boldsymbol{\mathcal{K}}_H(S) = \{K, P\}$, S ceases to be a separator in the reduced graph H^+. Since S is the critical separator of P in the graph H, and P contains subcritical separators, P has more than one maximal separator in the reduced graph H^+. Hence it is a non-leaf clique of H^+.

Conversely, since P is a leaf in H but not in H^+, the unique maximal separator of P in H ceases to be a separator when K is deleted from H. By Lemma 5.1, S is the only separator removed from $\boldsymbol{\mathcal{M}}_H$ when K is deleted, and it follows that $S = Sep_H(P)$. We also have $P = Sim_H(P) \cup Sep_H(P)$ by the remarks following Proposition 3.4. Now we claim that S cannot belong to any clique other than K and P. For, if it did belong to some other clique K' in the graph H, then we have $P \cap K' = S$; and $S = P \cap K'$ would continue to be the unique maximal separator of P in the reduced graph H^+. This implies that P is a leaf of H^+, contrary to supposition. Thus $\boldsymbol{\mathcal{K}}_H(S) = \{K, P\}$. If we order the separators in $\boldsymbol{\mathcal{S}}_H(P)$ as in (4.1), then we must have $S_1 = S$ and $\ell = 1$, since P has more than one maximal separator in the reduced graph H^+. It follows that S is the critical separator of P in H, and that P contains subcritical separators. □

The clique elimination scheme described in Figure 5.1 considers cliques from the set $\boldsymbol{\mathcal{L}}_G$ one by one. When a leaf clique K_j is eliminated, and $S = Sep_{H_{j-1}}(K_j)$ ceases to be a separator in the reduced graph H_j, three phenomena may occur. First, nonsimplicial vertices in a clique P that contains S may become newly simplicial, as described in the second part of Lemma 5.2. Second, if S happens to be one of exactly two incomparable maximal separators contained in a non-leaf clique P, then P now has a unique maximal separator and hence becomes a leaf in the reduced graph. Note that our elimination scheme does not include such a new leaf clique P in *P-leaves*, and hence P will not be a candidate for elimination. Third, if S is also the critical separator of a leaf clique P and the other conditions in Lemma 5.3 are satisfied, then P becomes a non-leaf in the reduced graph. In this case, P is removed from the set *P-leaves* and will not be considered for elimination.

We use the example in Figure 2.1 to illustrate some of these phenomena. If $Sim(K_1) = \{k_1\}$ is eliminated from G, then K_1 is no longer a maximal clique, and gets deleted from G. Now in the clique K_2 the vertex s_1 becomes simplicial; further S_1 ceases to be a separator and K_2 becomes a non-leaf in the reduced graph since it contains two maximal incomparable separators S_2 and S_3. If we eliminate $Sim(K_3) = \{k_3\}$ from this reduced graph, then one of these separators, S_2, ceases to be a separator, s_3 becomes a simplicial vertex in K_2, and K_2 becomes a leaf of the succeeding reduced graph.

Each eliminated clique K_j belongs to $\mathcal{L}_G$ and continues to be a leaf of the successive reduced graphs $H_1, \ldots, H_{j-1}$ until it is eliminated. Hence we call these cliques *persistent leaf cliques* and the elimination scheme that we have described is a *persistent leaf elimination scheme*. A clique $P \in \mathcal{L}_G$ that becomes a non-leaf when some clique K_j is eliminated from a reduced graph H_{j-1} will be called a *transilient leaf*. (We prefer *transilient*, denoting a sudden change in state, to *transient*, which means passing quickly into and out of existence. In this situation, a leaf changes state to a non-leaf clique, but continues to exist as a clique in the reduced graph.) The partition of the leaf cliques in $\mathcal{L}_G$ into persistent and transilient leaves is not unique, but depends on the order in which leaves are chosen for elimination.

6. A refined persistent leaf elimination scheme. We now incorporate two refinements into the persistent leaf elimination scheme of the previous section.

Refinement 1:

Eliminate the persistent leaves in non-increasing order of leaf separator sizes in the current graph H.

We organize the cliques in $\mathcal{L}_G$ into lists such that all leaves with the same leaf separator size are included in a list; if a leaf should become a transilient leaf during the scheme, then it is removed from this list; also, if the size of the leaf separator changes during the scheme, then the leaf is deleted from the list it belongs to, and then reinserted into the correct list.

Refinement 2:

Order the simplicial vertices in each maximal clique in queue order.

Thus if a vertex v becomes simplicial before another vertex w in a clique K, then v is eliminated before w when the clique K is chosen for elimination. This ordering of simplicial vertices is maintained for leaf as well as non-leaf cliques in G.

This scheme is shown in Figure 6.1. We will prove in the next section that this scheme computes a T-set of the graph G, and that these vertices are ordered in a TEO. In this section we characterize the set of persistent leaves and the set of vertices eliminated by the scheme.

Consider the persistent leaf scheme applied to the chordal graph G in Figure 2.1. The scheme would first eliminate K_1 or K_2 since these leaves have the maximum leaf separator size. If it eliminates K_1, then K_2 becomes a non-leaf in the reduced graph and is removed from the set *P-leaves*. Then the scheme would eliminate the persistent leaves K_3 and K_4, in any order, since both have the same leaf separator size. Suppose that K_3 is eliminated before K_4. Then the vertices are eliminated in the order k_1, k_3,

{initializations}
$P\text{-}leaves \leftarrow \mathcal{L}_G$;
for $K \in P\text{-}leaves$ **do** $Elim(K) \leftarrow Sim_G(K)$; **end for**
$H \leftarrow G$; $\hat{R} \leftarrow \emptyset$;
{eliminate simplicial and neosimplicial vertices from persistent leaves
by non-increasing leaf separator size}
while $P\text{-}leaves \neq \emptyset$ **do**
    Select $K \in P\text{-}leaves$ with maximum $|Sep_H(K)|$ for elimination;
    $\hat{R} \leftarrow \hat{R} \cup Elim(K)$;
    Choose $P \in \mathcal{K}_H$ for which $P \cap K = Sep_H(K)$;
    $H^+ \leftarrow H \setminus Elim(K)$ (in queue order);
    $P\text{-}leaves \leftarrow P\text{-}leaves - \{K\}$;
    if $P \in P\text{-}leaves$ **then**
        Append to $Elim(P)$ the vertices in $Sim_{H^+}(P) - Sim_H(P)$;
        if $P \notin \mathcal{L}_{H^+}$ **then**
            {P is a transilient leaf}
            $P\text{-}leaves \leftarrow P\text{-}leaves - \{P\}$;
        end if
    end if
    $H \leftarrow H^+$;
end while

FIG. 6.1. *A scheme for eliminating the persistent leaf cliques of G. We show later that the vertices eliminated form a T-set $\hat{R}$ of G and are ordered in a TEO of $G(\hat{R})$.*

k_4, and the simplicial vertices in K_2 are ordered as either $Elim(K_2) = \{k_2, s_1, s_3, s_2, s_4\}$ or $Elim(K_2) = \{k_2, s_1, s_3, s_4, s_2\}$.

We will find it useful to employ the concept of critical separators introduced earlier. Recall that $C(K)$ denotes the critical separator of a leaf clique K in the chordal graph G, and $\mathcal{C} = \{C_1, \ldots, C_p\}$ denotes the set of critical separators in G. The cliques in $\mathcal{L}_G$ are partitioned into p *cohorts* such that all leaves with critical separator C_i form a cohort $\mathcal{L}(C_i)$.

LEMMA 6.1. *If $\mathcal{C}$ contains only one critical separator, then the persistent-leaf elimination scheme eliminates all cliques in $\mathcal{L}(C) = \mathcal{L}_G$. Otherwise, this scheme eliminates all but one clique from every cohort such that $\mathcal{L}(C) = \mathcal{K}_G(C)$, and all cliques from every cohort such that $\mathcal{L}(C) \subset \mathcal{K}_G(C)$.*

Proof. From Lemma 5.3, a clique $P \in \mathcal{L}_G$ becomes a non-leaf when a leaf K is eliminated from a reduced graph H only if (i) $\mathcal{K}_H(C) = \{K, P\}$, (ii) C is the critical separator of P, and (iii) P contains subcritical separators. It follows that K and P are the only two uneliminated leaves in the cohort $\mathcal{L}(C)$.

If $\mathcal{C}$ contains only one critical separator C, then all leaves contain the separator C. We consider two cases, depending on whether G contains a non-leaf clique or not.

If G has a non-leaf clique, then we claim that every non-leaf clique of G contains C as well. For, consider a clique tree T of G. Then each non-leaf clique lies on a path between some pair of leaves K and K' of T. The leaves have the separator C in common; hence from the clique intersection property, every clique on the path from K to K', and in particular the non-leaf clique being considered, contains C as well. This proves the claim. When the penultimate leaf containing C is eliminated, the remaining leaf and the non-leaf clique(s) contain C, and condition (i) in Lemma 5.3 is violated. Hence the remaining leaf persists as a leaf in the reduced graph, and is eliminated by the scheme.

Now consider the case when all the cliques of G are leaves. Then since all leaves contain C, $K \cap K' \supseteq C$ for any pair of cliques K and K'. By the characterization of separators in Proposition 3.2, since the vertices in any separator form the intersection of a pair of cliques, there are no subcritical separators in any leaf. Then condition (iii) in Lemma 5.3 cannot be satisfied, and none of the leaves can become non-leaves in any of the successive reduced graphs. This completes the proof of the first statement.

If $\mathcal{C} = \{C_1, \ldots, C_p\}$ contains more than one separator, renumber the separators such that if $C_i \supset C_j$ then $i < j$, for every distinct pair $1 \leq i, j \leq p$. We prove the second statement by induction on k, the index of the critical separator. Consider the base case $k = 1$.

We consider first the subcase when $\mathcal{L}(C_1) = \mathcal{K}_G(C_1)$, and hence no non-leaf clique of G contains C_1. Now it is easily verified that the clique intersection graph of the connected chordal graph G is connected. Since $\mathcal{C}$ contains more than one critical separator, there is a clique K' outside $\mathcal{L}(C_1)$ that is adjacent to some clique K in the cohort such that (K, K') is an edge in some clique tree of G. Then by Proposition 3.2, $S = K \cap K'$ is a separator contained in K and K'. Now if $S \supseteq C_1$ then K' contains C_1, and this contradicts the condition that $\mathcal{L}(C_1) = \mathcal{K}_G(C_1)$; thus $S \subset C_1$. It follows that S is a subcritical separator in every clique in $\mathcal{L}(C_1)$.

From the definition, C_1 could be the critical separator of a clique K in $\mathcal{L}(C_1)$ because of three possibilities: either a non-leaf clique contains C_1, or C_1 is properly contained in all other separators in K, or no separator properly contained in C_1 contains all other separators contained in C_1. We have ruled out the first two possibilities in the previous paragraph, and hence in the graph G every clique in $\mathcal{L}(C_1)$ contains subcritical separators.

Now any leaf belonging to $\mathcal{L}(C_1)$ is either eliminated or becomes a transilient leaf when it is considered for elimination by the persistent leaf elimination scheme. By conditions (i) and (ii) in Lemma 5.3, a leaf in $\mathcal{L}(C_1)$ can become a transilient leaf only when the critical separator C_1 ceases to be a separator in the reduced graph. Thus all except the last leaf P to be considered for elimination in $\mathcal{L}(C_1)$ must be eliminated by the scheme. When K, the penultimate clique containing C_1, is eliminated by the scheme, since $\mathcal{L}(C_1) = \mathcal{K}_G(C_1)$, conditions (i) and (ii) in Lemma 5.3 are satisfied. We now show that condition (iii) is also satisfied, and hence that P becomes a transilient leaf. In the graph G the clique P contains subcritical separators from the argument in the preceding paragraph. By Lemma 5.1 the only way a separator can disappear during the elimination process is when it is the leaf separator of a leaf. Since the persistent leaf elimination scheme eliminates the leaves in non-increasing order of leaf separator sizes, any leaves whose leaf separators are properly contained in C_1 are not eliminated until all cliques in $\mathcal{L}(C_1)$ have been processed. Hence P continues to contain subcritical separators in the reduced graph obtained when K is eliminated. Thus we conclude that P, the sole remaining clique in the cohort $\mathcal{L}(C_1)$, becomes a transilient leaf.

If $\mathcal{L}(C_1) \subset \mathcal{K}_G(C_1)$, then by the renumbering of the critical separators there does not exist a leaf clique containing C_1 outside $\mathcal{L}(C_1)$ by Lemma 4.2. Hence there is a non-leaf clique containing C_1. Thus when the penultimate leaf in $\mathcal{L}(C_1)$ is eliminated, condition (i) in Lemma 5.3 is not satisfied. We conclude that the last clique in $\mathcal{L}(C_1)$ persists as a leaf in the reduced graph. Hence all cliques in $\mathcal{L}(C_1)$ are eliminated by the persistent leaf elimination scheme.

Now we consider the inductive step for $\mathcal{L}(C_k)$.

If $\mathcal{L}(C_k) = \mathcal{K}_G(C_k)$, then no other clique outside this cohort contains C_k. Now, an argument similar to the corresponding situation in the base case proves the result. It remains to consider the situation when $\mathcal{L}(C_k) \subset \mathcal{K}_G(C_k)$. If $\mathcal{K}_G(C_k)$ includes a non-leaf clique of G, the result follows from a similar argument as in the base case. If it does not, but includes a cohort $\mathcal{L}(C_j)$ such that $C_j \supset C_k$ (then $j < k$ by the ordering of the critical separators), consider the least index j satisfying the containment relation. We must have $\mathcal{L}(C_j) = \mathcal{K}_G(C_j)$, else there would exist a non-leaf clique containing C_k. By the order in which the leaves are considered for elimination, cliques in $\mathcal{L}(C_j)$ have been processed by the elimination scheme. By the inductive hypothesis, the last clique considered for elimination in $\mathcal{L}(C_j)$ has become a transilient leaf when C_j ceased to be a separator. Since this clique contains C_k, all of the cliques in $\mathcal{L}(C_k)$ are eliminated by the persistent leaf elimination scheme. $\Box$

The example graph G has three critical separators; $\mathcal{L}(S_1) = \mathcal{K}_G(S_1) = \{K_1, K_2\}$; $\mathcal{L}(S_2) = \{K_3\}$, $\mathcal{L}(S_3) = \{K_4\}$, and the latter two cohorts are properly contained in the set of cliques containing their critical separators. Hence we conclude from the lemma that a persistent leaf set could be either $\{K_1, K_3, K_4\}$ or $\{K_2, K_3, K_4\}$.

It is instructive to compare the set of persistent leaves with the largest set of leaves eliminated by a shortest clique tree algorithm designed by Blair and Peyton [5]. They organize the leaves into cohorts such that all leaves with the same *leaf separator* belong to one cohort. They showed that their algorithm chooses all but one of the leaves from a cohort where $\mathcal{L}(S) = \mathcal{K}_G(S)$, and all the leaves from a cohort satisfying $\mathcal{L}(S) \subset \mathcal{K}_G(S)$.

We now characterize the vertices eliminated by the persistent leaf elimination scheme. We consider three subsets of the maximum cardinality T-set R: R^N, the subset of R belonging to some non-leaf clique of G; R^T, the subset belonging to the transilient leaves of G; and R^P, the subset *eliminated* from the persistent leaves of G. We will show that these subsets partition R.

By Lemma 3.6 the subset R^N consists of simplicial vertices of G belonging to the non-leaf cliques. The other two subsets could include simplicial as well as neosimplicial vertices of G. The next result states that all vertices of R that belong only to persistent leaf cliques are eliminated by the persistent leaf elimination scheme.

Lemma 6.2. *The three subsets R^N, R^T, and R^P partition the maximum cardinality T-set R. Furthermore, $R^N \cup R^T$ is a set of simplicial vertices in the graph $G^+ = G \setminus R^P$.*

Proof. If v is a simplicial vertex belonging to a persistent leaf clique of G, then it belongs to no other clique, and hence is eliminated by the elimination scheme. It is also clear that simplicial vertices belonging to non-leaves and transilient leaves are not eliminated by the scheme. Hence consider what happens to a neosimplicial vertex v.

By the characterization in Theorem 4.1, $\boldsymbol{\mathcal{K}}_G(v) \subseteq \boldsymbol{\mathcal{L}}_G$, and in every clique $K \in \boldsymbol{\mathcal{K}}_G(v)$, v belongs only to the supercritical separators of K. By Lemma 4.3, v belongs to a subset of the cliques in some unique cohort $\boldsymbol{\mathcal{L}}(C_j)$.

If all cliques of G belong to the cohort $\boldsymbol{\mathcal{L}}(C_j)$, then by Lemma 6.1 all cliques are eliminated by the persistent leaf elimination scheme, and $v \in R^P$. If the cliques of G belong to more than one cohort, then again by Lemma 6.1, either all cliques in $\boldsymbol{\mathcal{L}}(C_j)$ are eliminated, or all but one are eliminated. Since v belongs only to supercritical separators in the cliques in $\boldsymbol{\mathcal{L}}(C_j)$, when the penultimate clique containing the last supercritical separator that includes v is eliminated, v becomes simplicial in the only uneliminated clique that contains it. This last clique could be either a persistent leaf or a transilient leaf, but not a non-leaf of G, since $\boldsymbol{\mathcal{K}}_G(v) \subseteq \boldsymbol{\mathcal{L}}_G$. If it is the former, then $v \in R^P$, and if it is the latter, then $v \in R^T$. □

In the example, if we assume that the cliques K_1, K_3, and K_4 are eliminated, then $R = \{k_1, \ldots, k_4, s_1\}$; $R^P = \{k_1, k_3, k_4\}$; $R^T = \{k_2, s_1\}$; and $R^N = \emptyset$.

7. T-sets. In this section we prove that the vertices eliminated by the persistent leaf elimination scheme form a T-set $\hat{R}$ of the graph G, and that the vertices are eliminated in a TEO of $G(\hat{R})$. (These concepts are defined in Section 2.)

Theorem 7.1. *The set of vertices R^P eliminated by the persistent leaf elimination scheme is a T-set of G; furthermore, the scheme eliminates these vertices in a TEO of $G(R^P)$.*

Before we can prove this theorem we need two auxiliary results. We omit the proof of the following lemma since it is similar to the proof of Theorem 3.2 in [18].

Lemma 7.2. *Let $\hat{R}$ be a T-set of G. An ordering α of $G(\hat{R})$ is a TEO of $G(\hat{R})$ if and only if for every $u, v \in \hat{R}$ such that $u \in adj^-(v)$, $\alpha(u) < \alpha(v)$.* □

Lemma 7.3. *A vertex set $\hat{R} \subseteq R$ is a T-set of G if and only if $adj^-(v) \subset \hat{R}$ for every vertex $v \in \hat{R}$.*

Proof. We first prove the 'only if' part by contraposition. Assume that there exists a vertex $v \in \hat{R}$ such that $u \notin \hat{R}$ for some vertex $u \in adj^-(v)$. We need to prove that there exists no TEO of $G(\hat{R})$. It suffices to show that any PEO of $G(\hat{R})$ cannot be a TEO of $G(\hat{R})$. Let α be a PEO of $G(\hat{R})$. Since $v \in \hat{R}$, but $u \notin \hat{R}$, we have $\alpha(v) < \alpha(u) = n + 1$. By Lemma 2.2 there exists a vertex $w \in adj^-(v)$ that is not adjacent to u. For α to be a PEO of $G(\hat{R})$, we must have $\alpha(w) < \alpha(v) < \alpha(u) = n+1$. Now $(w, v), (v, u) \in E$ and $(w, u) \notin E$, and thus it follows that α is not a TEO of $G(\hat{R})$.

To prove the 'if' part, choose a vertex set $\hat{R} \subseteq R$ such that $adj^-(v) \subset \hat{R}$ for every vertex $v \in \hat{R}$. We observe that because every set $adj^-(v)$ ($v \in \hat{R}$) is contained in $\hat{R}$, there exists an ordering α of $G(\hat{R})$ satisfying the following property: for every $u, v \in \hat{R}$ such that $u \in adj^-(v)$, we have $\alpha(u) < \alpha(v)$. If α is any such ordering of $G(\hat{R})$, then by Lemma 7.2, α is a TEO of $G(\hat{R})$. Consequently $\hat{R}$ is a T-set of G. □

Proof of Theorem 7.1:
We make use of the characterization of a T-set in Lemma 7.3, showing that if $v \in R^P$ then for every $u \in adj_G^-(v)$, $u \in R^P$. If $v \in R^P$ is simplicial in G, then $adj^-(v)$ is the empty set, and there is nothing to prove. Hence consider a neosimplicial vertex $v \in R^P$. We have $\lambda_G(v) = 2$, and for all $u \in adj_G^-(v)$, $\lambda_G(u) \leq 2$.

Let $\{K_1, \ldots, K_p\}$ be the set of persistent leaves eliminated by the scheme, listed in the order in which they are eliminated. Denote $G \equiv H_0$, and for $j = 1, \ldots, p$, let $H_j = H_{j-1} \setminus Sim_{H_{j-1}}(K_j)$ be the reduced graph obtained from H_{j-1} by eliminating current simplicial vertices in the clique K_j. We denote $G^+ \equiv H_p$, the final reduced graph when all the persistent leaf cliques in G have been eliminated. Recall that the simplicial vertices in each clique K are maintained in a queue $Elim(K)$ to which vertices are added in the order in which they become simplicial.

Since v is a neosimplicial vertex that belongs to R^P, v is a simplicial vertex in some reduced graph H_j. At this juncture in the elimination process all but one of the cliques in $\boldsymbol{\mathcal{K}}_G(v)$ have been eliminated. Now by Lemma 3.1 $u \in adj_G^-(v)$ implies that $\boldsymbol{\mathcal{K}}_G(u) \subset \boldsymbol{\mathcal{K}}_G(v)$. Hence when v becomes simplicial in H_j, either all the cliques in $\boldsymbol{\mathcal{K}}_G(u)$ have been eliminated, or u and v belong to the sole remaining clique from $\boldsymbol{\mathcal{K}}_G(v)$. In the former case, u has been eliminated when v becomes simplicial, and hence is ordered before v. We now consider the latter case.

In this situation, the vertices u and v are both simplicial vertices belonging to the same clique K in H_j. Hence we need to show that u appears before v in the queue $Elim(K)$. If $\lambda_G(u) = 1$, then u is simplicial, and since v is not simplicial in G, the result holds. Now consider $\lambda_G(u) = 2$. Let $S_1 \supset S_2 \ldots \supset S_r = C(K)$ denote the supercritical separators in K. Since $\boldsymbol{\mathcal{K}}_G(u) \subset \boldsymbol{\mathcal{K}}_G(v)$, the vertex u belongs to the separators S_1, $\ldots$, S_i, and v to S_1, $\ldots$, S_j, where $i < j \leq r$. The consequence of eliminating cliques in nondecreasing order of leaf separator sizes is that S_j ceases to be a separator in a reduced graph before S_i does. Hence u becomes simplicial in the clique K before v does. This completes the proof. □

8. A greedy leaf elimination scheme and its optimality. The following two sections constitute the third part of the paper, where we develop a greedy leaf clique elimination scheme based on the results in the previous two parts, prove that it solves Problem 1, and then describe an efficient implementation.

Let G_{i+1} denote the reduced graph obtained from a chordal graph G_i by using the scheme in Fig. 6.1 to eliminate a set of persistent leaf cliques. Let $G \equiv G_1$, $\ldots$, G_t, $G_{t+1} = \emptyset$ be the sequence of chordal graphs obtained by repeatedly applying this scheme to the original chordal graph G. We call this a *greedy leaf clique elimination scheme* since it eliminates a largest set of cliques it can delete from the graph at each step. In this section we prove that this scheme solves Problem 1 by comparing it with the greedy vertex elimination scheme described in [18]. At each step both schemes are shown to identify the same set of newly simplicial and neosimplicial vertices since the lengths of relevant vertices are the same in the reduced graphs obtained in the two schemes.

LEMMA 8.1. *Let K be a non-leaf clique in a chordal graph G, and let $G^+ = G \setminus R^P$*

denote the reduced graph obtained by the elimination of a set of persistent leaf cliques of G. If v is a vertex of the graph $G^* = G^+ \setminus Sim_G(K)$, then $\lambda_{G^+}(v) = \lambda_{G^*}(v)$.

Proof. We make use of the following observations about chordless paths in the proof of this lemma and the next one. A chordless path cannot have more than two vertices from a clique, since a third vertex creates a chord. Further, there is a clique which contains any two consecutive vertices on the path. A simplicial vertex must be an endpoint of a chordless path. Finally, a longest chordless path cannot increase in length as simplicial vertices are eliminated.

The clique K could be a non-leaf or a leaf clique of G^+. There are two cases to consider.

Case 1: K is a non-leaf of G^+.

The elimination of $Sim_G(K)$ does not change the separators or the set of maximal cliques in G^+, by Lemma 5.1. Hence a simplicial vertex in G^* is also a simplicial vertex in G^+, and consequently the lemma holds for all vertices with length one. We now consider vertices with length greater than one.

A chordless path in G^+ which includes none of the vertices in $Sim_G(K)$ continues to be a chordless path in G^*. A vertex in $Sim_G(K)$ is a simplicial vertex in G^+, and hence is an endpoint of every chordless path to which it belongs to in G^+. If both end-points of a chordless path in G^+ belong to $Sim_G(K)$, then the path cannot include any other vertices, since the only vertices adjacent to simplicial vertices in K belong to K. Hence consider a chordless path $[u, v, w, \ldots]$ in G^+ with $u \in Sim_G(K)$. Then v belongs to some separator S' in K, and w to some clique K' such that $K \cap K' = S'$. Since the path is chordless, indeed $w \in K' - S'$. The clique K is a non-leaf in G^+, and thus K contains another separator S'' such that $S'' - S'$ is not empty. We replace u on the path by a vertex $t \neq v$ belonging to $S'' - S'$. No edge joins t to a vertex on this path other than v since the separator S' separates vertices in $K - S'$ from $K' - S'$. Thus we have replaced a chordless path in G^+ containing a vertex from $Sim_G(K)$ by a chordless path in G^* without changing its length.

Case 2: K is a leaf of G^+.

The facts that K is a non-leaf in G and a leaf in G^+ imply that K becomes a leaf when some persistent leaf K_j is eliminated from a reduced graph H_{j-1} resulting in another reduced graph H_j. Since K is a non-leaf in H_{j-1}, K contains two incomparable maximal separators S, S' in this graph. When K_j is eliminated, one of these separators, say S', ceases to be a separator in H_j, and S remains as the leaf separator of K in H_j. Now there exists a vertex $s' \in S' - S$ since the two separators are incomparable. This vertex s' cannot belong to any other clique in the reduced graph H_j because vertices in a leaf clique are partitioned into the subset of simplicial vertices, which belong to no separator, and the subset of vertices belonging to the leaf separator. It follows that s' belongs only to K in H_j, and is simplicial in this graph. Further, s' continues to be simplicial in G^+ since vertices of K are not eliminated by the persistent leaf elimination scheme, K being a non-leaf clique of G. Note also that s' is a nonsimplicial vertex of G, and hence does not belong to $Sim_G(K)$.

The consequence of this latter observation is that K continues to be a maximal clique in the graph G^* when vertices in $Sim_G(K)$ are eliminated from G^+. Thus a simplicial vertex of G^* is also simplicial in G^+, and the lemma holds for all vertices with $\lambda(.) = 1$. Hence we consider vertices with $\lambda(.) > 1$.

As in the previous subcase, if both end-points of a chordless path in G^+ belong to $Sim_G(K)$, then this path contains no other vertices. Hence consider the case when only one endpoint belongs to $Sim_G(K)$. Let $u \in Sim_G(K)$, and let $v \notin Sim_G(K) \cup \{s'\}$ be a vertex belonging to K with $\lambda(v) > 1$. Then any chordless path $[u, v, \ldots,]$ in the graph G^+ cannot include the simplicial vertex s'. Furthermore, we can replace u by s' in the above path without changing its length in the graph G^+. The latter is a path in the reduced graph G^*, and the result follows. □

LEMMA 8.2. *Let $G^+ = G \setminus R^P$ denote the reduced graph obtained when a set of persistent leaf cliques are eliminated from G, and let K be a transilient leaf. Further, let $G^* = G^+ \setminus Elim(K)$, where $Elim(K)$ contains the set of simplicial vertices in K when it becomes a non-leaf. If v is a vertex in G^*, then $\lambda_{G^+}(v) = \lambda_{G^*}(v)$.*

Proof. The case when K is a non-leaf in G^+ can be treated exactly as in the previous lemma. Hence assume that K is a leaf in G^+. In our notation, the persistent leaves eliminated (in order) from G are $K_1, \ldots, K_p$, and H_j is the reduced graph obtained when simplicial vertices in K_j are eliminated from a graph H_{j-1}, where $H_0 \equiv G$, and $H_p \equiv G^+$.

The clique K is a leaf in the initial graph H_0, becomes a non-leaf in some reduced graph H_i, and is again a leaf in the final reduced graph H_p. (It is possible that K cycles between being a non-leaf and a leaf a few times.) Denote by H_j the highest-numbered reduced graph in which K became a non-leaf when a persistent leaf K_j was eliminated from the reduced graph H_{j-1}. It later became a leaf again when some other persistent leaf K_ℓ (where $j < \ell \leq p$) was eliminated from a reduced graph $H_{\ell-1}$. Thus K contained two or more maximal incomparable separators in H_j and in every successive reduced graph until H_ℓ. Again, as in Lemma 8.1, K contains a vertex s', which belonged to one of the maximal separators of K in H_j but which does not belong to the leaf separator of K in H_ℓ. This vertex is simplicial in H_ℓ, but was a nonsimplicial vertex in H_j, and hence it does not belong to $Elim(K)$. Now we can repeat the rest of the argument in the previous lemma for the case when K is a leaf to show that the length of a vertex v belonging to G^* is the same in the graphs G^+ and G^*. □

THEOREM 8.3. *Let $R = R^P \cup R^N \cup R^T$ be a partition of the maximum cardinality T-set of the graph G. If $G^+ = G \setminus R^P$, $G^* = G \setminus R$, and v is a vertex belonging to G^*, then $\lambda_{G^+}(v) = \lambda_{G^*}(v)$.*

Proof. We can conclude from the previous two lemmas that the elimination of $R^T \cup R^N$ does not create any new simplicial vertices in G^*. Hence the theorem is true for all vertices with length one, and we proceed to consider vertices of length greater than or equal to two.

By Lemma 6.2, $R^N \cup R^T$ is a set of simplicial vertices in G^+. Hence any chordless path in G^+ containing these vertices must include them as end-points of the path. If an endpoint of a chordless path in G^+ belongs to $R^N \cup R^T$, then from the proofs of Lemmas 8.1 and 8.2, we can replace it by a vertex from the same clique but not belonging to $R^N \cup R^T$, without changing its length in the graph G^+. We can do this independently for each of the two end-points. (For, if both end-points belong to the same clique, then the path must have length one. Then both vertices belong to $R^N \cup R^T$, and do not belong to G^*.) The resulting path belongs to G^*, and thus the result follows. □

THEOREM 8.4. *The greedy leaf clique elimination scheme obtains a minimum cardinality T-partition of the chordal graph G.*

Proof. We will prove that the greedy leaf elimination scheme obtains a T-partition of G with the same number of T-sets in it as the greedy vertex elimination scheme in [18] that is known to be optimal. As above, let G^+ (G^*) denote the reduced graph obtained from G by eliminating the vertices in R^P (R). From the characterization of R in terms of lengths in Theorem 2.3, and from Theorem 8.3, we have

$$R_{G^+} = R_{G^*} \cup R^N \cup R^T.$$

Thus the T-set of G^+ identified by the greedy leaf elimination scheme includes the maximum cardinality T-set of the reduced graph G^*, and a few additional simplicial vertices that do not belong to G^*. Furthermore, the above observation can be made inductively with respect to each successive nonempty graph $G_2, \ldots, G_t$ in the sequence of chordal graphs generated by the greedy vertex elimination scheme.

In the greedy vertex elimination scheme G_{t+1} is the empty graph. We need to show that the greedy leaf elimination scheme eliminates all the vertices in the reduced graph G' it considers at the t-th step. (From the preceding two paragraphs, this is G_t augmented by some simplicial vertices.) Since G_{t+1} is the empty graph, all vertices in G' are simplicial or neosimplicial. Hence by Lemma 3.6, all cliques of G' are leaves, and by Theorem 4.1, there are no subcritical separators in any of the leaves. Then by Lemma 5.3, the greedy leaf elimination scheme eliminates all the cliques of G'. This completes the proof. □

The greedy vertex elimination scheme applied to the graph in Figure 2.1 yields $R_1 = \{k_1, k_2, s_1, k_3, k_4\}$; $R_2 = \{s_3, s_2, s_4\}$. Assuming that K_1, K_3, and K_4 are eliminated, the greedy leaf elimination scheme results in $\hat{R}_1 = \{k_1, k_3, k_4\}$; $\hat{R}_2 = \{k_2, s_1, s_3, s_2, s_4\}$. In both cases, the vertices are listed in a compound TEO of G.

9. An implementation of the persistent leaf elimination scheme. We describe briefly an implementation of the persistent leaf elimination scheme. We discuss the clique tree data structure and a simple test used to identify leaf cliques in the reduced graphs before describing the algorithm.

The only representation of the chordal graph G needed is a rooted clique tree of G, defined in [14], and computed from a PEO of G. The rooted clique tree T from [14] has the following important property: If C is a child of a clique K in the clique tree T, then there exists a vertex $v \in K \cap C$ such that $v \notin A$, where A is any clique that is not a descendant of K in T.

In this section we will need to distinguish between a leaf of the clique tree T and a leaf of the chordal graph G. Recall that the leaves of T form a subset of the leaves of G. A clique that is not a leaf of T will be called an *interior clique*, whereas we have already denoted a clique that is not a leaf of G by a non-leaf clique. In the rooted clique tree T, the children C of each interior clique K are initially sorted by nondecreasing order of the intersection $|K \cap C|$.

When a leaf clique K of G is eliminated, the rooted clique tree T is updated as described in [14] to represent the resulting reduced graph. If K is also a leaf of T, then the only update necessary is to delete K from T. If K is an interior clique of T, then the update is more involved. The first child C_1 of K is 'promoted' to the place of K, and C_1 becomes the parent of the other children of K. The latter cliques are listed in their current order after the existing children of C_1.

9.1. Identifying a leaf clique. Let $\mu(K) \equiv |Sim_H(K)|$ be the number of simplicial vertices in a clique K, and let $\pi(K)$ be the size of a largest separator contained in K. The following result (Blair and Peyton [5]) is immediate from Proposition 3.4.

PROPOSITION 9.1. *A clique K is a leaf of a chordal graph H if and only if $\mu(K) + \pi(K) = |K|$; it is a non-leaf if and only if $\mu(K) + \pi(K) < |K|$.*

Updating the number of simplicial vertices in a clique during elimination is an easy matter. Updating the size of a largest separator during elimination is more involved. However, we claim that it suffices to maintain and update the size of *any* maximal separator in a clique K instead of $\pi(K)$ to identify it as leaf or a non-leaf in a reduced graph. If K is a non-leaf, since the size of any maximal separator is no greater than $\pi(K)$, when the former size is used instead of the latter, the test in Proposition 9.1 will identify it as a non-leaf. If K is a leaf, then it has a unique maximal separator, and hence, again, the test suffices.

We choose the size of a particular maximal separator with respect to the rooted clique tree. Let

$$\iota(K) = \begin{cases} |K \cap C_1|, & \text{if } K \text{ is an interior clique of } T, \\ |K \cap P|, & \text{if } K \text{ is a leaf of } T, \end{cases}$$

where C_1 is the first child of an interior clique K, and P is the parent of a leaf K. The quantity $\iota(K)$ is easily updated when a leaf or an interior clique of T is eliminated during the algorithm.

We need to prove that $\iota(K)$ is the size of a maximal separator of K in each reduced graph. This is trivial if K is a leaf of the clique tree T, and hence we establish it when K is an interior clique. The separator $K \cap C_1$ is a maximal separator of K in the initial clique tree because (i) for every child C, the intersection $K \cap C$ contains a vertex that does not belong to $K \cap P$, and (ii) by the initial ordering of the children of K, $|K \cap C_1|$ is at least as large as any other separator size $|K \cap C|$. If K acquires a new child D after an interior clique elimination, and C is a child of K before the update, then $K \cap C$ contains a vertex that does not belong to $K \cap D$ ([14], proof of Theorem 12). Hence $K \cap C_1$ continues to be a maximal separator of K in the reduced graph. We have thus proved the following result.

PROPOSITION 9.2. *Let H denote a chordal graph at any stage in the persistent leaf elimination algorithm, and let $\iota(K)$ be defined as described above with respect to a* *rooted clique tree T of H. Then $\mu(K)+\iota(K)=|K|$ if and only if K is a leaf of H; and $\mu(K)+\iota(K)<|K|$ if and only if K is a non-leaf of H.* □

Input: A clique tree T representing a chordal graph G; the simplicial vertices of each $K \in \boldsymbol{\mathcal{K}}_G$ organized in a queue $Elim(K)$.
Output: Upon termination, $\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_t$ is a minimum-cardinality T-partition, where each partition member $\hat{R}_i$ is the T-set R^P belonging only to a set of persistent leaves in the reduced graph $G_i = G \setminus \{\hat{R}_1 \cup \cdots \cup \hat{R}_{i-1}\}$.

$G_1 \leftarrow G$; $H \leftarrow G$; $\hat{R}_1 \leftarrow \emptyset$; $i \leftarrow 1$; $P\text{-}leaves \leftarrow \boldsymbol{\mathcal{L}}_G$;
while $G_i \neq \emptyset$ **do**
  while $P\text{-}leaves \neq \emptyset$ **do**
    Let K be a clique with the largest leaf separator size in $P\text{-}leaves$;
    Delete K from $P\text{-}leaves$;
    $\hat{R}_i \leftarrow \hat{R}_i \cup Elim(K)$ (in queue order); $H^+ \leftarrow H \setminus Elim(K)$;
    Choose $P \in \boldsymbol{\mathcal{K}}_H$ such that $P \cap K = Sep_H(K)$;
    Insert newly simplicial vertices in $Sep_H(K)$ into the tail of $Elim(P)$;
    Update the clique tree to reflect the elimination of K;
    if $P \in P\text{-}leaves$ **then**
      if P is a non-leaf in H^+ **then**
        delete P from $P\text{-}leaves$;
      else $\{P$ is a leaf in $H^+\}$
        update the leaf separator size of P;
      end if
    end if
    $H \leftarrow H^+$;
  end while
  $G_{i+1} \leftarrow G_i \setminus \hat{R}_i$; $\hat{R}_{i+1} \leftarrow \emptyset$; $P\text{-}leaves \leftarrow \boldsymbol{\mathcal{L}}_{G_{i+1}}$; $i \leftarrow i+1$;
end while

FIG. 9.1. *Algorithm to compute a minimum cardinality T-partition.*

9.2. Algorithm. A clique tree algorithm for computing a minimum cardinality T-partition is shown in Figure 9.1. The algorithm eliminates a set of persistent leaf cliques of the chordal graph G_i (maintained in the set *P-leaves*) during the ith iteration (*major step*) of the outermost **while** loop.

To facilitate processing the leaf cliques by non-increasing order of separator sizes, the leaf cliques in *P-leaves* can be organized into *lists* such that $list(\iota)$ includes all leaves K with $\iota(K)=\iota$. Similarly, new leaves eligible for consideration at the next major step are included in lists $newlist(.)$. A leaf is deleted from its list if it becomes a non-leaf, or if its leaf separator size changes. In the latter case, it is inserted into the correct list. To maintain efficient insertion and deletion during the algorithm, each list is doubly linked. This enables us to find a persistent leaf with the largest leaf separator size efficiently.

We proceed to analyze the complexity of the algorithm. Recall that $|V| = n$, m is the number of maximal cliques of G, and $q = \sum_{K \in \mathcal{K}_G} |K|$ is the size of the clique tree

It is shown in [14] that the elimination of a clique K and the update of the clique tree can be performed in $|Elim(K)| + |Sep_H(K)| = |K|$ time. We consider the work done in maintaining and examining *list* and *newlist* separately. All other work within the second **while** loop in Figure 9.1 requires $\mathcal{O}(|K|)$ time, and hence over the course of the algorithm is bounded by $\mathcal{O}(m + q)$.

Now we consider the work necessary to manipulate *list*. We observe that a leaf clique whose leaf separator size is maximum is eliminated from the reduced graph G_i by the algorithm, since the first leaf processed for elimination cannot become a non-leaf. For convenience denote this leaf by K_i. The work done to process the leaves of G_i in *P-leaves* is $\mathcal{O}(p_i + \iota_{max})$, where $p_i \equiv |P\text{-}leaves|$, and $\iota_{max} \equiv |Sep_{G_i}(K_i)|$. The second part of this cost, summed over the algorithm, is $\sum_{i=1}^{t} |Sep_{G_i}(K_i)|$, which is clearly bounded by $\mathcal{O}(q)$.

To establish a bound on the first part of the above cost, we need to bound the number of times a leaf clique P in *P-leaves* becomes a non-leaf or changes its leaf separator size. The key observation is that any of these phenomena occurs when a leaf clique K from *P-leaves* is eliminated, causing the leaf separator in P to cease being a separator in the reduced graph. Thus the total number of such transitions over the entire algorithm is bounded by $m-1$. Hence the first part of the cost, summed over the algorithm, is bounded by the number of cliques eliminated, m, and the total number of transitions considered above, $m - 1$. Thus this cost is also $\mathcal{O}(m)$.

It only remains to consider the cost of creating and updating *newlist* when persistent leaves in G_i are being eliminated. If we charge the cost of identifying P as a leaf in G_{i+1} and inserting into the correct *newlist*(.) to the persistent leaf K being eliminated, this cost is $\mathcal{O}(m)$ over the algorithm.

We conclude that the time complexity of the algorithm is $\mathcal{O}(n + q)$. The space complexity is easily seen to satisfy the same bound as well.

10. Concluding remarks. We now discuss the class of chordal graphs that have transitive perfect elimination orderings; i.e., graphs for which Problem 1 has the solution $R_1 = V$. From (2.1) such graphs have $\lambda(v) \leq 2$ for all $v \in V$. Hence this is the class of "P_4-free" chordal graphs, i.e., chordal graphs that do not contain an induced P_4 (the path on four vertices). A P_4-free graph can be characterized in terms of its clique intersection graph: All of its separators can be linearly ordered with respect to the set containment relation. Then all of its cliques are leaves, and the smallest separator is the only critical separator in the graph. This class has been studied in earlier work by Wolk (arborescence comparability graphs [8]), and Golumbic (trivially perfect graphs [10]).

The class of P_4-free chordal graphs is related to other interesting subclasses of chordal graphs.

An *s-trampoline* T_s is graph on $2s$ vertices $X \cup Y$ such that $X = \{x_1, \ldots, x_s\}$ is a clique, $Y = \{y_1, \ldots, y_s\}$ is an independent set, and x_1, y_1, x_2, y_2, ..., x_s, y_s, x_1 is a Hamiltonian cycle. A graph G is *strongly chordal* if it is chordal and, for any $s \geq 3$, does not contain an s-trampoline as an induced subgraph [7]. P_4-free chordal graphs form a proper subset of strongly chordal graphs. This can be seen from the fact that when $s \geq 3$, every T_s contains an induced P_4.

A *threshold graph* is a chordal graph that does not contain an induced P_4 or $2K_2$ (a pair of independent edges) [11]. It follows that the class of P_4-free chordal graphs properly contains threshold graphs.

P_4-free chordal graphs are interesting from the perspective of sparse matrix computations as well. An important issue here is the relation between fill and parallelism in Cholesky factorization. The height of the elimination tree can be used as a simple measure of the number of steps needed to factor the matrix in parallel. It is well-known that for the path on n vertices, if no fill is permitted, the shortest elimination tree has height $\lceil n/2 \rceil$, while there exists an elimination tree of height $\log_2 n$ if $\mathcal{O}(n)$ fill is permitted. Hence the question: For what classes of graphs does increasing fill not lead to increased parallelism in sparse Cholesky factorization?

The problem of computing a vertex ordering that leads to a shortest elimination tree is NP-complete for an arbitrary graph [19]. If G is chordal, then Liu [15] has shown that a scheme due to Jess and Kees that recursively eliminates a maximum independent subset of the simplicial vertices computes a shortest elimination tree of G *over all PEOs of* G. If G is a P_4-free chordal graph, then it has the property that in every induced subgraph of G, the size of a maximum independent set of vertices (MIS) is equal to the number of maximal cliques in the subgraph. Further, every clique in the induced subgraph is a leaf, and hence contains a simplicial vertex; thus a maximum independent set consisting of simplicial vertices is also a MIS in every induced subgraph of G. Hence it turns out that for a P_4-free chordal graph G, the Jess and Kees scheme computes a shortest elimination tree over *all* vertex orderings of G. Thus permitting additional fill cannot lead to increased parallelism for these graphs. One expects a similar result to hold for most "real-life" problems as well, though the proof of such a result would be hard to obtain.

Acknowledgment. The third author would like to thank Professor Joseph Liu for the guidance and encouragement he received when he was a student at York University.

REFERENCES

[1] F. L. Alvarado, A. Pothen, and R. S. Schreiber, *Highly parallel sparse triangular solution.* In [9], pp. 141–158.

[2] F. L. Alvarado and R. S. Schreiber, *Optimal parallel solution of sparse triangular systems*, SIAM J. Sci. Comput., 14 (1993), pp. 446–460.

[3] C. Berge, *Graphs*, North Holland, Amsterdam, second edition, 1985.

[4] P. A. Bernstein and N. Goodman, *The power of natural semijoins*, SIAM J. Comput., 10 (1981), pp. 751–771.

[5] J. R. S. Blair and B. W. Peyton, *On finding minimum diameter clique trees*, Tech. Report TM/11850, Oak Ridge National Lab., Oak Ridge, TN, 1991.

[6] ——, *An introduction to chordal graphs and clique trees*. In [9], pp. 1–30.

[7] G. J. Chang, M. Farber, and Z. Tuza, *Algorithmic aspects of neighborhood numbers*, SIAM J. Disc. Math., 6 (1993), pp. 24–29.

[8] P. Duchet, *Classical perfect graphs*, in Topics on Perfect Graphs (Ann. Discr. Math., 21), C. Berge and V. Chvatal, eds., Amsterdam, 1984, North Holland, pp. 67–96.

[9] J. A. George, J. R. Gilbert, and J. W. H. Liu, eds., *Graph Theory and Sparse Matrix Computation*, Springer Verlag, 1993. (IMA volumes in Mathematics and its Applications, Vol. 56.)

[10] M. C. Golumbic, *Trivially perfect graphs*, Discr. Math., 24 (1978), pp. 105–107.

[11] ——, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[12] N. J. Higham and A. Pothen, *The stability of the partitioned inverse approach to parallel sparse triangular solution*, SIAM J. Sci. Comput., 15 (1994), pp. 139–148.

[13] C. W. Ho and R. C. T. Lee, *Counting clique trees and computing perfect elimination schemes in parallel*, Inf. Proc. Lett., 31 (1989), pp. 61–68.

[14] J. G. Lewis, B. W. Peyton, and A. Pothen, *A fast algorithm for reordering sparse matrices for parallel factorization*, SIAM J. Sci. Stat. Comput., 6 (1989), pp. 1146–1173.

[15] J. W. H. Liu, *Reordering sparse matrices for parallel elimination*, Parallel Comput., 11 (1989), pp. 73–91.

[16] M. Lundquist, *Zero patterns, chordal graphs, and matrix completions*, PhD thesis, Clemson University, Clemson, SC, 1990.

[17] B. W. Peyton, *Some applications of clique trees to the solution of sparse linear systems*, PhD thesis, Clemson University, Clemson, SC, 1986.

[18] B. W. Peyton, A. Pothen, and X. Yuan, *Partitioning a chordal graph into transitive subgraphs for parallel sparse triangular solution*, Lin. Alg. Applic., 192 (1993), pp. 329–354.

[19] A. Pothen, *The complexity of optimal elimination trees*, Tech. Report CS-88-16, Computer Science, Pennsylvania State University, 1988.

[20] A. Pothen and F. L. Alvarado, *A fast reordering algorithm for parallel sparse triangular solution*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 645–653.