On some simplicial elimination schemes for chordal graphs

Michel Habib, Vincent Limouzy

▶ **To cite this version:**

Michel Habib, Vincent Limouzy. On some simplicial elimination schemes for chordal graphs. 2008. hal-00353959v2

HAL Id: hal-00353959
https://hal.science/hal-00353959v2

Preprint submitted on 17 Feb 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On some simplicial elimination schemes for chordal graphs

Michel Habib [1]

LIAFA, CNRS and Université Paris Diderot - Paris 7, France

Vincent Limouzy [2]

Dept. of Computer Science, University of Toronto, Canada

Abstract

We introduced here an interesting tool for the structural study of chordal graphs, namely the **Reduced Clique Graph**. Using some of its combinatorial properties we show that for any chordal graph we can construct in linear time a simplicial elimination scheme starting with a pending maximal clique attached via a minimal separator maximal under inclusion among all minimal separators.

Keywords: Chordal graphs, minimal separators, simplicial elimination scheme, reduced clique graph.

1 Introduction

In the following text, a graph is always finite, simple, loopless, undirected and connected. A graph is **chordal** iff it has no chordless cycle of length ≥ 4. The class of chordal graphs is one of the first class to have been studied at

[1] Email: habib@liafa.jussieu.fr
[2] Email: limouzy@cs.toronto.edu

the beginning of the theory of perfect graphs. Since then chordal graphs have been intensively studied, as can be seen in the following books [9,2].

Let us recall the main notions defined for chordal graphs. A **maximal clique** of G is a complete subgraph maximal under inclusion. A **minimal separator** is a subset of vertices S for which it exist $a, b \in G$ such that a and b are not connected in $G-S$, and S is minimal under inclusion with this property. A vertex is **simplicial** if its neighborhood is a clique (complete graph). An ordering $x_1, \ldots, x_n$ of the vertices is a **simplicial elimination scheme**, if for every $i \in [1, n-1]$ x_i is a simplicial vertex in $G[x_{i+1}, \ldots x_n]$. A **maximal clique tree** is a tree T that satisfies the following three conditions: Vertices of T are associated with the maximal cliques of G. Edges of T correspond to minimal separators. For any vertex $x \in G$, the cliques containing x yield a subtree of T.

Using results of Dirac [5], Fulkerson, Gross [6], Buneman [3], Gavril [8] and Rose, Tarjan and Lueker [12], we have:

Theorem 1.1 *The following 5 statements are equivalent and characterize chordal graphs.*

(i) *G has a simplicial elimination scheme*

(ii) *Every minimal separator is a clique*

(iii) *G admits a maximal clique tree.*

(iv) *G is the intersection graph of subtrees in a tree.*

(v) *Any LexBFS provides a simplicial elimination scheme.*

2 The Reduced Clique Graph

Definition 2.1 For a chordal graph G, we denote by $\mathcal{C}$ the set of maximal cliques of G and by $\mathcal{C}_r(G)$ the **reduced clique graph**, i.e. the graph whose vertices are the maximal cliques of G, and two cliques are joined by an edge iff their intersection separates them (i.e. if for every $x \in C - (C \cap C')$ and every $y \in C' - (C \cap C')$, $C \cap C'$ is a minimal separators for x and y in G).

Clearly $\mathcal{C}_r(G)$ is a subgraph of the intersection graph of the maximal cliques of G. Each edge CC' of $\mathcal{C}_r(G)$ can be labelled with the minimal separator $S = C \cap C'$.

Lemma 2.2 *[7] Let us consider three maximal cliques C_1, C_2, C_3 in G, such that $S = C_1 \cap C_2$ and $U = C_2 \cap C_3$ are minimal separators in G, then $S \subset U$ implies that $C_1 \cap C_3$ is a minimal separator of G.*

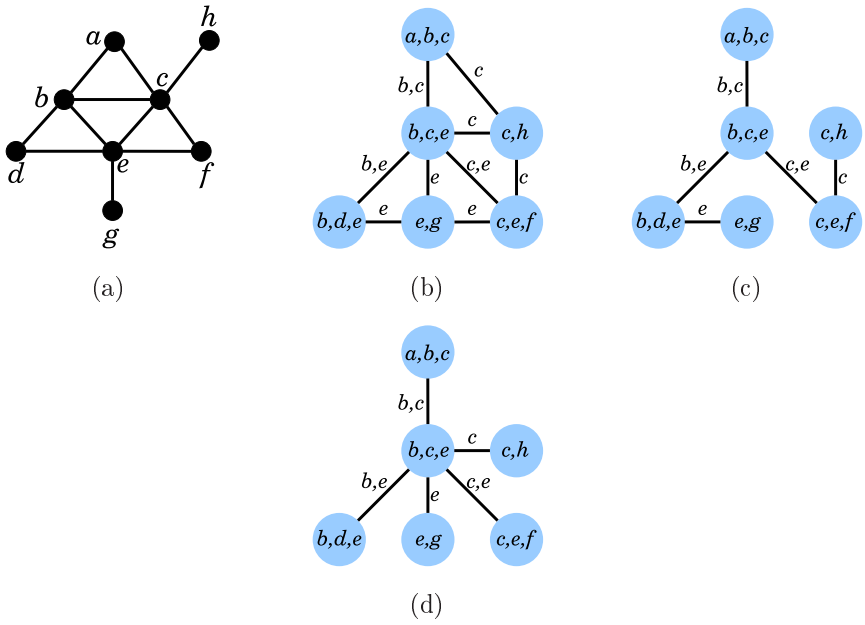


Fig. 1. An example of a chordal graph (a), its reduced clique-graph (b), note that although the maximal cliques $\{b, d, e\}$ and $\{c, e, f\}$ intersect the corresponding edge is missing. Two maximal clique-trees are shown (c)-(d).

Lemma 2.3 *[7] Let us consider a triangle in $\mathcal{C}_r(G)$ together with its 3 minimal separators labelling its edges. Then two of these minimal separators must be equal and included in the third.*

With these two lemmas it is easy to prove the following result:

Proposition 2.4 *[1,7] For a chordal graph G maximal clique trees correspond to maximum spanning trees of $\mathcal{C}_r(G)$ when the edges are labelled with the size of the minimal separator they are associated with. Furthermore $\mathcal{C}_r(G)$ is the union of all maximal clique trees of G.*

As a consequence, all maximal clique trees define the same multiset of minimal separators, and from one maximal clique tree to another we can proceed by exchanging edges (with same label) on triangles. But the graph $\mathcal{C}_r(G)$ has still more combinatorial properties, that we now consider. Let us now study the limit case of the two previous lemmas, when $S = U$. First we need a basic separating lemma (which can also be found in a more general setting of tree decompositions, see lemma 12.3.1 in [4]).

Lemma 2.5 *Separating lemma*

Let T be a maximal clique tree and C_1C_2 and edge of T. Let T_1 and T_2 the

two connected components of $T - C_1C_2$. If we define V_i for i=1,2 the union of all maximal cliques in T_i. Then $S = C_1 \cap C_2$ separates every $x \in V_1 - S$ from any $y \in V_2 - S$.

Lemma 2.6 *Let us consider three maximal cliques C_1, C_2, C_3 in G, such that $S = C_1 \cap C_2 = U = C_2 \cap C_3$ are minimal separators in G, then either the edge $C_1C_3 \in \mathcal{C}_r(G)$ or the two edges C_1C_2, C_2C_3 cannot belong both to a same maximal clique tree.*

Proof. Suppose that the edge C_1C_3 does not belong to $\mathcal{C}_r(G)$, i.e. that $S = C_1 \cap C_3$ does not separate $C_1 - S$ from $C_3 - S$. Therefore if it exists some maximal clique tree T containing both edges C_1C_2, C_2C_3, this would contradict the above separating lemma 2.5. □

Lemma 2.7 *Let us consider three maximal cliques C_1, C_2, C_3 in G, such that $S = C_1 \cap C_2 = U = C_2 \cap C_3$ are minimal separators in G, if the edges C_1C_2, C_2C_3 belong both to a same maximal clique tree T. Then $C_1C_3 \in \mathcal{C}_r(G)$ and $C_1 \cap C_3 = U$*

Proof. Using the previous lemma necessarily $C_1C_3 \in \mathcal{C}_r(G)$, but lemma 1 just states that $C_1 \cap C_3 \subseteq U = S$. If this is a strict inclusion then one can build a new maximal clique tree T' by exchanging the edges C_1C_2 by C_1C_3. But then T' would be a better spanning tree than T which contradicts the optimality of T and therefore $C_1 \cap C_3 = U = S$. □

3 Min-max separators

For a finite chordal graph G, let us call a min-max (resp. min-min) separator S, a minimal separator that is maximal (resp. minimal) under inclusion among all minimal separators of G.

Theorem 3.1 *[10] Let G be a chordal graph, then it exists a maximal clique-tree T that admits a pending edge labelled with a min-max separator.*

Proof. The proof will proceed by transforming a maximal clique tree using the above lemmas. Let us consider T a maximal clique tree of G and some edge $ab \in T$ labelled with a min-max separator S. First we need to define an operation on cliques trees, namely the **chain-reduction**. Suppose ab is not a pending edge in T, therefore $T - \{ab\}$ is the disjoint union of two non empty trees T_a, T_b. If one of these trees, say T_a admits a lead edge xy labelled with a minimal separator $S' \subset S$ (y being the pending clique in T). Then the whole chain in T_a joining ab to xy is labelled with minimal separators containing

S'. Using this fact and successive applications of the above lemmas, we can interchange in T_a the edges xy and ay (or equivalently in T exchanging xy by by). Let us go back to the proof of the theorem. If one of the subtrees T_a, T_b, say T_a is made up with edges labelled with minimal separators included in S, then using the chain-reduction operation we can produce another maximal clique tree T' in which all the edges of T_a are leaves attached to b and ab is a leaf and we have finished. Else it exists in one of the subtrees T_a, T_b, say T_a, some edge zt labelled with S' which is not comparable with S. We recurse on the maximal minimal separator that contains S' and which necessarily belongs to T_a. This process necessarily ends by finding a leaf in the tree which is labelled with a max-min separator, because each time we recurse on a strict subtree. □

Such maximal clique trees seem to play an important role for the study of path graphs [10]. The above proof also suggests a dual result for min-min separators. But as it was noticed by M. Preissmann [11], such a maximal clique tree does not always exist. The graph depicted in figure 2 does not admit a min-min elimination scheme.

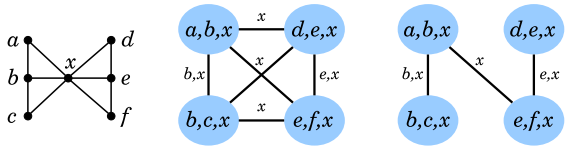


Fig. 2. Preissmann's counter example [11], A graph, its reduced clique graph and one maximal clique tree

Using the above constructive proof, a polynomial scheme can be obtained to compute a min-max elimination schemes. As shown in Figure 3, classical graph searches do not provide such elimination scheme.

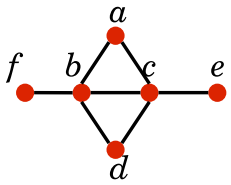


Fig. 3. An exemple of graph on which MCS, $LexBFS$ fail to find a max-min simplicial vertex. For any starting vertex, both searches will end on e of f.

Corollary 3.2 *Such trees can be obtained in linear time.*

Proof. We prove the result in the min-max case. To obtain such a tree we can first compute a maximal clique tree T of G as explained in [7], with its edges being labelled with the minimal separators of G. We can sort the minimal separators with respect to their size in linear time, and therefore start with an edge ab labelled with a max-min separator S and then explore T_a and stop either because the whole subtree is labelled with minimal separators contained in S, then it suffices to modify the tree, or because we have found an edge labelled with some edge xy labelled with a minimal separator S' incomparable with S. In this case, among all edges in T_a, consider the edge zt labelled with a min-max separator S'' incomparable with S, and recurse on zt. During this algorithm an edge of T is at most traversed twice, which yields the linearity of the whole process. □

Corollary 3.3 *For any chordal graph there exist an elimination scheme that follows a linear extension of the containment ordering of the minimal separators. It can be computed in $O(n.m)$.*

Proof. It is well-known, that one can produce elimination scheme on the following way. Take any maximal clique tree T of a chordal graph G, and let C be a leaf of this tree, attach to the tree via the minimal separator S. Successively prune all vertices in $C - S$ and recurse on $T - C$ the maximal clique tree of $G - \{C - S\}$. To finish the proof it suffices to apply the above theorem. Each time the above algorithm is applied requires $O(n + m)$, this yields the complexity. □

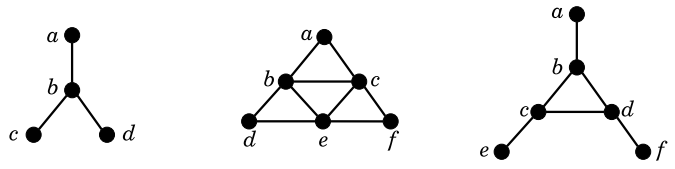
It should be noticed that not every linear extension of the containment ordering can be obtained with an elimination scheme.

4 Reversible elimination schemes

A reversible elimination scheme is just an ordering of the vertices which is simplicial in both directions. As shown by the graph called 3-sun, there exist graphs for which one can prove that there is no reversible elimination scheme. A vertex is said to be **bisimplicial** if its neighbourhood can be partionned into two cliques. Furthermore, if a graph G admits such a reversible elimination scheme, this implies that each vertex is either simplicial or bisimplicial. Therefore such a graph cannot contain any claw ($K_{1,3}$) as subgraph.

Theorem 4.1 *A graph G admits a reversible ordering if and only if G is proper interval graph.*

Proof. Let us consider a unit interval graph G and one of its unitary interval representation. Therefore to each vertex $x \in G$ we can associate an interval $I(x) = [left(x), right(x)]$ of length one of the real line, such that xy is an edge iff $I(x) \cap I(y) \neq \emptyset$. Let us consider the total ordering τ of the vertices of G defined as follows: $x \leq_\tau y$ iff $(right(x) < right(y))$. Let x be the first vertex of this ordering, clearly its neighborhood is a clique. Thus τ is an elimination scheme. Reversibility is straightforward. Conversely let us proceed by contradiction. Let us assume that G admits a reversible elimination ordering and that G is not a proper interval graph. As proper interval graph admit a characterization by forbidden induced subgraphs, we can assume that our graph contains one of the graph as a subgraph. The forbidden sugraphs for proper interval graphs are the net, the claw and the sun of size 3. These graphs are depicted in figure 4. So to prove our claim it is sufficient to see that none of these graphs admit a reversible elimination ordering. For the claw, we already noticed it. Considering the 3-sun, it is easy to check that each vertex is bisimplicial. If we consider the sugraph induced by $\{a, b, c, d, e\}$, this graph forms the bull. And this graph admit only one reversible elimination ordering which is a, b, d, c, e. To convince ourself a and e has to be the extremities of the ordering (d is not a good candidate since it is not simplicial in the whole graph). Then to satisfy b, since a is already positionned c and d have to be on the right. In the same way to satisfy c, since e is already positionned b and d have to be on the left. Finally the only ordering to fullfill all the constraints is a, b, d, c, e. But now, when we want to add f, each position in the previous order will violate the constraint for at least one vertex. A contradiction. For the net, the proof is similar. □



(a) Claw: $K_{1,3}$ (b) 3-Sun (c) *net*

Fig. 4. Forbidden induced subgraphs for proper interval graphs.s

Acknoledgements:

We are grateful to B. Lévêque for pointing out useful references.

References

[1] J.R.S. Blair and B. Peyton. An introduction to chordal graphs and clique trees. *Graph Theory and Sparse Matrix Multiplication*, pages 1–29, 1993.

[2] A. Brandstädt, V.B. Le, and J.P. Spinrad. *Graph Classes: A Survey*. SIAM Monographs on Disc. Math. and Applic., 1999.

[3] P. Buneman. A characterization of rigid circuit graphs. *Discrete Math.*, 9:205–212, 1974.

[4] R. Diestel. *Graph Theory*. Springer Verlag, 1997.

[5] G.A. Dirac. On rigid circuit graphs. *Abh. Math. Sem. Univ. Hamburg*, 25:71–76, 1961.

[6] D.R. Fulkerson and O.A. Gross. Incidence matrices and interval graphs. *Pacific J. of Math.*, 15:835–855, 1965.

[7] P. Galinier, M. Habib, and C. Paul. Chordal graphs and their clique graphs. In Springer-Verlag, editor, *21th Workshop on Graph-Theoretic Concepts in Computer Science, Aachen, Lecture Notes in Computer Science 1017*, pages 358–371, 1995.

[8] F. Gavril. The intersection graphs of a path in a tree are exactly the chordal graphs. *J. Combinatorial Theory*, 16:47–56, 1974.

[9] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

[10] B. Lévêque, F. Maffray, and M. Preissmann. Characterizing path graphs by forbidden induced subgraphs. *Journal of Graph Theory*, To appear, 2008.

[11] M. Preissmann, 2009. private communication.

[12] D.J. Rose, R.E. Tarjan, and G.S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM J. of Computing*, 5:266–283, 1976.