Paper III

Lex M versus MCS-M

Yngve Villanger*

Abstract

We study the problem of minimal triangulation of graphs. One of the first algorithms to solve this problem was Lex M, which was presented in 1976. A new algorithm, and a simplification of Lex M called MCS-M, was presented in 2002. In this paper we compare these two algorithms and show that they produce the same set of triangulations, answering an open question mentioned by the authors of MCS-M.

1 Introduction

Graph theory has several important problems that involve creating a chordal supergraph from a given graph by adding a set of edges. The set of added edges is called *fill*, and the chordal supergraph is called a *triangulation* of the given graph. Different goals may be desired; one is to introduce as few new edges as possible (called *minimum fill*), and another is to create a triangulation such that the largest clique is as small as possible, which corresponds to the *treewidth* of the graph. Both of these problems are NP-hard [1, 13].

Minimal fill, also called *minimal triangulation*, is the problem of adding an inclusion minimal set of fill edges. There exist several practical algorithms that solve this problem [2, 4, 5, 6, 7, 9, 11, 12]. Since minimum fill is hard to compute, minimal fill may be used as an alternative, even though the difference in the number of fill edges may be quite large. One of the algorithms that solve the minimal triangulation problem is Lex M (Rose, Tarjan, and Lueker [12]), which is a classical algorithm based on a special breadth first search and lexicographic labeling of the vertices. Recently (Berry, Blair, Heggernes, and Peyton [3]) introduced a new algorithm called MCS-M. This is a simplification of Lex M so that cardinality weights are used instead of lexicographic labels.

A triangulation of a graph can also be obtained by using the *elimination game* [10] algorithm. This algorithm takes a graph and an ordering of the vertices as input. The ordering of the vertices given to the elimination game is also called

*Department of Informatics, University of Bergen, N-5020 Bergen, Norway. yngvev@ii.uib.no

an *elimination ordering.* This ordering uniquely defines the set of fill edges for a given graph, but there may be many different elimination orderings that introduce the same set of fill edges. If an ordering produces a minimal triangulation, then the ordering is called a *minimal elimination ordering* (meo).

Both Lex M and MCS-M produce an meo. The user of Lex M and MCS-M can select the last vertex in the ordering, and may have some choices during the execution of the algorithm. Because of these choices, both algorithms produce a set of minimal orderings for a given graph. Some of the orderings may only occur in one of the sets, and it follows that the number of orderings in each of these sets can be quite different. However, in this paper we show that for every Lex M ordering, there exists an MCS-M ordering that creates exactly the same fill edges, and for every MCS-M ordering there exists a Lex M ordering that creates exactly the same fill edges. It follows that Lex M and MCS-M create exactly the same set of triangulations.

2 Elimination orderings, Lex M, and MCS-M

We consider finite, simple, undirected and connected graphs. Given a graph $G = (V, E)$, we denote the number of vertices as $n = |V|$ and the number of edges as $m = |E|$. The *neighborhood* of a vertex $u \in V$ is denoted by $N_G(u) = \{v$ for $(u, v) \in E\}$, and $N_G[u] = N_G(u) \cup \{u\}$. In the same way we define the neighborhood of a set $A \subseteq V$ of vertices by $N_G(A) = \cup_{u \in A} N_G(u) \backslash A$. A sequence $v_1 - v_2 - ... - v_k$ of distinct vertices describes a *path* if (v_i, v_{i+1}) is an edge for $1 \leq i < k$. The *length* of a path is the number of edges in the path. A *cycle* is defined as a path except that it starts and ends with the same vertex. If there is an edge between every pair of vertices in a set $A \subseteq V$, then the set A is called a *clique.*

Chordal graphs are the family of graphs where every cycle of length greater than three has a chord. A *chord* is an edge between two non-consecutive vertices of a cycle. Chordal graphs can be computed from non-chordal graphs by introducing new edges, called *fill* edges. This process is called *triangulation* of a graph. An *ordering* of V is a function $\alpha : \{1, 2, ..., n\} \leftrightarrow V$, and we use $\alpha = [v_1, v_2, ..., v_n]$ to denote that $\alpha(i) = v_i$ for $1 \leq i \leq n$. Given a graph G and an ordering α of the vertices in G, the *elimination game* [10] can be used to obtain a triangulation G^+_α of the given graph G. The triangulation is obtained by picking the first vertex from the ordering, making its neighborhood into a clique, and then removing the vertex from the graph. This is repeated until no vertex remains. The ordering α is called an *elimination ordering.* The vertex at position i is given by $\alpha(i)$, and $\alpha^{-1}(u)$ gives us the position of the vertex u in the ordering. Theorem 2.1 gives a precise description of what edges exist in the resulting graph.

Theorem 2.1 *(Rose, Tarjan, and Lueker [12]) Given a graph $G = (V, E)$ and an elimination ordering α of G, (y, z) is an edge in G^+_α if and only if $(y, z) \in E$ or there exists a path $y, x_1, x_2, ..., x_k, z$ in G where $\alpha^{-1}(x_i) < min\{\alpha^{-1}(y), \alpha^{-1}(z)\}$, for $1 \leq i \leq k$.*

The set of vertices *monotonely adjacent* to a vertex is the set of higher numbered neighbors, and is defined as follows. Given a graph $G = (V, E)$ and an ordering α of the vertices, then $madj_{G^+_\alpha}(z) = \{w$ for which $(z, w) \in E(G^+_\alpha), \alpha^{-1}(z) < \alpha^{-1}(w)\}$. Our first result, before we continue with minimal triangulations, concerns changes that can be done to an elimination ordering without altering the resulting triangulation. Our approach is to consider two consecutive vertices in the ordering, and decide if they can switch places in the ordering without altering the triangulation.

Lemma 2.2 *Given a graph $G = (V, E)$, and an ordering $\alpha = [x_1, x_2, ..., x_k, u, v, x_{k+3}, ..., x_n]$ of $V(G)$, let $\beta = [x_1, x_2, ..., x_k, v, u, x_{k+3}, ..., x_n]$ (u and v are swapped). If $(u, v) \notin E(G^+_\alpha)$ then $G^+_\alpha = G^+_\beta$.*

Proof. We want to show that $madj_{G^+_\alpha}(z) = madj_{G^+_\beta}(z)$ for each $z \in V$, since it then follows that $G^+_\alpha = G^+_\beta$. Let z be any vertex in $V \setminus \{u, v\}$. The set of vertices appearing prior to z in α and in β is exactly the same. It follows from Theorem 2.1 that $madj_{G^+_\alpha}(z) = madj_{G^+_\beta}(z)$ for any $z \in V \setminus \{u, v\}$. Let us now consider the vertices u and v. The edge $(u, v) \notin E(G^+_\alpha)$, and due to Theorem 2.1, there exists no path from u to v in G that passes through only vertices from among $x_1, x_2, ..., x_k$. We show that $madj_{G^+_\alpha}(u) = madj_{G^+_\beta}(u)$. In order to do this we will show that both $madj_{G^+_\alpha}(u) \setminus madj_{G^+_\beta}(u)$ and $madj_{G^+_\beta}(u) \setminus madj_{G^+_\alpha}(u)$ are empty sets. Let us first on the contrary assume that there exists a vertex $z \in madj_{G^+_\beta}(u) \setminus madj_{G^+_\alpha}(u)$. Then there must exist a path from u to z in G that passes through only vertices from $x_1, x_2, ..., x_k, v$, and this path must contain v, since there does not exist any path in G between u and z that uses only vertices from $x_1, x_2, ..., x_k$, because $z \notin madj_{G^+_\alpha}(u)$. This gives a contradiction since there exists no path in G from u to v that only uses vertices from among $x_1, x_2, ..., x_k$, and thus no such path between u and z through v can exist. Now let us on the contrary assume that there exists a vertex $z \in madj_{G^+_\alpha}(u) \setminus madj_{G^+_\beta}(u)$. This is a contradiction since there must exist a path from u to z in G that passes through only vertices from $x_1, x_2, ..., x_k$, but no such path that passes through only vertices from $x_1, x_2, ..., x_k, v$. It follows that $madj_{G^+_\alpha}(u) = madj_{G^+_\beta}(u)$. It remains to show that $madj_{G^+_\alpha}(v) = madj_{G^+_\beta}(v)$. The proof is the same as the one for u. Let us first on the contrary assume that there exists a vertex $z \in madj_{G^+_\alpha}(v) \setminus madj_{G^+_\beta}(v)$. Then there must exist a path from v to z in G that passes through only vertices from $x_1, x_2, ..., x_k, u$. This path must contain u because there does not exist any path in G that passes through only vertices from $x_1, x_2, ..., x_k$, since $z \in madj_{G^+_\alpha}(v) \setminus madj_{G^+_\beta}(v)$. This is a contradiction since there does not exist a path from u to v in G, that passes through only vertices from $x_1, x_2, ..., x_k$. Let us on the contrary assume that there exists a vertex $z \in madj_{G^+_\beta}(v) \setminus madj_{G^+_\alpha}(v)$. This is a contradiction because there must exist a path from v to z in G that passes through only vertices from $x_1, x_2, ..., x_k$, but there must not exist any path in G passes through only vertices from $x_1, x_2, ..., x_k, u$, since $z \in madj_{G^+_\beta}(v) \setminus madj_{G^+_\alpha}(v)$. It follows that $madj_{G^+_\alpha}(v) = madj_{G^+_\beta}(v)$. ∎

Lex M computes a minimal elimination ordering given a graph. The elimination order is produced in reverse order, and in some implementations of Lex M, the highest-numbered vertex in the ordering can be selected arbitrarily by the user. Each vertex in Lex M is assigned a label. This label is a sequence of numbers ordered in decreasing order. Let $L(u)$ be the label of vertex u, and let $L_k(u)$ be the number at position k in the sequence $L(u)$. The labels can be compared in the following way: $L(u) = L(v)$ if $|L(u)| = |L(v)|$ and $L_i(u) = L_i(v)$ for $1 \leq i \leq |L(u)|$. Furthermore $L(u) < L(v)$ if $L_k(u) < L_k(v)$, where k is the smallest number such that $L_k(u) \neq L_k(v)$, or $L_i(u) = L_i(v)$ for $1 \leq i \leq |L(u)|$ and $|L(u)| < |L(v)|$.

Algorithm Lex M (Rose, Tarjan, and Lueker [12])
Input: $G = (V, E)$.
Output: A minimal elimination ordering α and G^+_α.

$G^+_\alpha = G$;
for all vertices u in G
    $L(u) = \emptyset$;
for $i = n$ to 1
    let v be one of the unnumbered vertices with largest label;
    $\alpha^{-1}(v) = i$;
    for each unnumbered vertex u such that there exists a path
    $u = x_0, x_1, ..., x_k = v$ in G, where x_j is unnumbered and
    $L(x_j) < L(u)$ for $0 < j < k$
        add i to $L(u)$;
        add fill edge (v, u) to G^+_α;

Just as Lex M does, MCS-M produces an elimination ordering in reverse order, and like Lex M the highest-numbered vertex in the ordering can be selected arbitrarily by the user in some implementations of MCS-M. MCS-M differs from Lex M by using cardinality weights instead of lexicographic labels. MCS-M basically uses the same approach as Lex M to search the graph.

Algorithm MCS-M (Berry, Blair, Heggernes, and Peyton [3])
Input: $G = (V, E)$.
Output: A minimal elimination ordering α and G^+_α.

$G^+_\alpha = G$;
for all vertices u in G
    $w(u) = 0$;
for $i = n$ to 1
    let v be one of the unnumbered vertices with largest weight;
    $\alpha^{-1}(v) = i$;
    for each unnumbered vertex u such that there exists a path
    $u = x_0, x_1, ..., x_k = v$ in G, where x_j is unnumbered and
    $w(x_j) < w(u)$ for $0 < j < k$
        $w(u) = w(u) + 1$;
        add fill edge (v, u) to G^+_α;

Both Lex M and MCS-M may provide the user with choices from the set of unnumbered vertices with largest label or weight, respectively. These choices are not necessarily the same for the two algorithms. In Figure 1, there is an example where Lex M and MCS-M do not have the same choices.

Figure 1: Let 2 be the starting vertex in the given graph. In this situation Lex M is capable of creating the following set of elimination orderings $[\{4, 3, 1, 2\}, \{4, 1, 3, 2\}]$, while MCS-M is capable of creating the following set of orderings $[\{4, 3, 1, 2\}, \{4, 1, 3, 2\}, \{1, 4, 3, 2\}]$. Observe that every one of these orderings is a perfect elimination ordering (peo) [8] for the given graph.

To make it easier to discuss Lex M and MCS-M we give an exact description of the label and weight for each vertex at each step of the algorithm. Let $L_{z-}(x)$ be the label of vertex x in Lex M right before z has been assigned the number $\alpha^{-1}(z)$, and let $L_{z+}(x)$ be the label of x right after z has been assigned the number $\alpha^{-1}(z)$ and Lex M has added this number to the labels described by Lex M. Lemma 2.3 describes how the relationship between labels changes as the algorithm proceeds.

Lemma 2.3 *(Rose, Tarjan, and Lueker [12]) Let $G = (V, E)$ be a graph, and let u, v be vertices of G. If $L_{\alpha(i)-}(v) < L_{\alpha(i)-}(u)$, then $L_{\alpha(j)-}(v) < L_{\alpha(j)-}(u)$ for all $1 \leq j \leq i$.*

For MCS-M we do the same, let $w_{z-}(x)$ be the weight of vertex x in MCS-M right before z has been assigned the number $\alpha^{-1}(z)$, and let $w_{z+}(x)$ be the weight of x right after z has been assigned the number $\alpha^{-1}(z)$ and MCS-M has used z to increase the weight of other vertices as described by MCS-M. Given a set $A \subseteq V$ of vertices, then $hW_{z-}(A)$ is the set of vertices in A with the highest weight assigned by MCS-M right before z has been assigned a number, and $hL_{z-}(A)$ is the set of vertices in A with the largest labels assigned by Lex M right before z has been assigned a number.

3 Labeling in Lex M

Lex M and MCS-M do a quite similar search along paths of unnumbered vertices, and use this to find the set of vertices of which they change the labels (resp. weight). An easy observation is that the length of a label in Lex M increases by exactly one every time Lex M changes it. We will now study the relation between the length and value of a pair of labels in Lex M, when there is an unnumbered path between the vertices containing the labels.

Lemma 3.1 *Assume that there is an unnumbered path $x_0, x_1, ..., x_k$ in G right before step $\alpha^{-1}(z)$ of Lex M, where $k \geq 1$, $u = x_0$, and $v = x_k$, and let $L_{z-}(x_i) \leq L_{z-}(u)$ where $0 < i < k$. Then $|L_{z-}(u)| > |L_{z-}(v)|$ if and only if $L_{z-}(u) > L_{z-}(v)$.*

Proof. ($\Rightarrow$) Let us first on the contrary assume that $|L_{z-}(u)| > |L_{z-}(v)|$ and $L_{z-}(u) \leq L_{z-}(v)$. Let u' be a vertex such that $\alpha^{-1}(u')$ is a number in $L_{z-}(u) \setminus L_{z-}(v)$, which does exist since $|L_{z-}(u)| > |L_{z-}(v)|$. It follows that $\alpha^{-1}(u') > \alpha^{-1}(z)$ since $\alpha^{-1}(u') \in L_{z-}(u)$. Let p be the largest number such that $0 \leq p < k$ and $\alpha^{-1}(u') \in L_{u'+}(x_p)$. We will show by contradiction that $L_{u'-}(x_p) \geq L_{u'-}(x_i)$ for $p < i \leq k$. Let q be the smallest number such that $p < q \leq k$ and $L_{u'-}(x_q) > L_{u'-}(x_p)$. Now we have a path $x_p, x_{p+1}, ..., x_{p+l} = x_q$, where $L_{u'-}(x_q) > L_{u'-}(x_j)$ for $p \leq j < p + l$, and since $\alpha^{-1}(u') \in L_{u'+}(x_p)$ there exists a path from x_p to u' where the labels of all intermediate vertices in the path are smaller than both the labels of x_p and u'. Thus we have a path from x_q to u', where every intermediate vertex has a smaller label than x_q and u'. This is a contradiction since $\alpha^{-1}(u') \notin L_{u'+}(x_q)$. Now we return to our main proof. Since $L_{u'-}(x_p) \geq L_{u'-}(x_i)$ and $\alpha^{-1}(u') \notin L_{u'+}(x_i)$ for $p < i \leq k$, while $\alpha^{-1}(u') \in L_{u'+}(x_p)$, we have $L_{u'+}(x_p) > L_{u'+}(x_i)$ for $p < i \leq k$. It follows from Lemma 2.3 that $L_{z-}(x_p) > L_{z-}(v)$, where $0 \leq p < k$, since $L_{u'+}(x_p) > L_{u'+}(v = x_k)$. Now we have a contradiction since we assumed that $L_{z-}(x_i) \leq L_{z-}(u)$ where $0 < i < k$ and that $L_{z-}(v) \geq L_{z-}(u)$.
($\Leftarrow$) Let us next on the contrary assume that $|L_{z-}(u)| \leq |L_{z-}(v)|$ and $L_{z-}(u) > L_{z-}(v)$. Let v' be a vertex such that $\alpha^{-1}(v')$ is a number in $L_{z-}(v) \setminus L_{z-}(u)$; such a vertex does exist since $|L_{z-}(v)| \geq |L_{z-}(u)|$ and $L_{z-}(v) < L_{z-}(u)$. It follows that $\alpha^{-1}(v') > \alpha^{-1}(z)$ since $\alpha^{-1}(v') \in L_{z-}(v)$. Let q be the smallest number such that $0 < q \leq k$ and $\alpha^{-1}(v') \in L_{v'+}(x_q)$. We will show by contradiction

that $L_{v'-}(x_q) \geq L_{v'-}(x_i)$ for $0 \leq i < q$. Let p be the largest number such that $0 \leq p < q$ and $L_{v'-}(x_p) > L_{v'-}(x_q)$. Now we have a path $x_p, x_{p+1}, ..., x_{p+l} = x_q$, where $L_{v'-}(x_p) > L_{v'-}(x_j)$ for $p < j \leq p+l$, and since $\alpha^{-1}(v') \in L_{v'+}(x_q)$ there exists a path from x_q to v' where the labels of all intermediate vertices in the path are smaller than both the labels of x_q and v'. Thus we have a path from x_p to v', where every intermediate vertex has a smaller label than x_p and v'. This is a contradiction since $\alpha^{-1}(v') \notin L_{v'+}(x_p)$. Now we return to our main proof. Since $L_{v'-}(x_q) \geq L_{v'-}(x_i)$ and $\alpha^{-1}(v') \notin L_{v'+}(x_i)$ for $0 \leq i < q$, while $\alpha^{-1}(v') \in L_{v'+}(x_q)$, we have $L_{v'+}(x_q) > L_{v'+}(x_i)$ for $0 \leq i < q$. It follows from Lemma 2.3 that $L_{z-}(x_q) > L_{z-}(u)$, where $0 < q \leq k$, since $L_{v'+}(x_q) > L_{v'+}(u = x_0)$. Now we have a contradiction since we assumed that $L_{z-}(x_i) \leq L_{z-}(u)$ where $0 < i < k$ and that $L_{z-}(u) > L_{z-}(v)$. ∎

Lemma 3.2 *Assume that there is an unnumbered path $x_0, x_1, ..., x_k$ in G right before step $\alpha^{-1}(z)$ of Lex M, where $k \geq 1$, $u = x_0$, and $v = x_k$, and let $L_{z-}(x_i) \leq L_{z-}(u)$ where $0 < i < k$. Then $|L_{z-}(u)| < |L_{z-}(v)|$ if and only if $L_{z-}(u) < L_{z-}(v)$.*

Proof. ($\Leftarrow$) Let us assume that $L_{z-}(u) < L_{z-}(v)$ and then prove that $|L_{z-}(u)| < |L_{z-}(v)|$. It follows that $L_{z-}(x_i) < L_{z-}(v)$ for $0 < i < k$ since $L_{z-}(u) < L_{z-}(v)$. Lemma 3.1 can now be used on the path $x_k, x_{k-1}, ..., x_0$ for $k \geq 1$ where $u = x_k, v = x_0$; thus $|L_{z-}(u)| < |L_{z-}(v)|$.
($\Rightarrow$) Let us on the contrary assume that $|L_{z-}(u)| < |L_{z-}(v)|$ and $L_{z-}(u) \geq L_{z-}(v)$. Let v' be a vertex such that $\alpha^{-1}(v')$ is a number in $L_{z-}(v) \backslash L_{z-}(u)$; such a vertex does exist since $|L_{z-}(v)| > |L_{z-}(u)|$. It follows that $\alpha^{-1}(v') > \alpha^{-1}(z)$ since $\alpha^{-1}(v') \in L_{z-}(v)$. Let q be the smallest number such that $0 < q \leq k$ and $\alpha^{-1}(v') \in L_{v'+}(x_q)$. We will show by contradiction that $L_{v'-}(x_q) \geq L_{v'-}(x_i)$ for $0 \leq i < q$. Let p be the largest number such that $0 \leq p < q$ and $L_{v'-}(x_p) > L_{v'-}(x_q)$. Now we have a path $x_p, x_{p+1}, ..., x_{p+l} = x_q$, where $L_{v'-}(x_p) > L_{v'-}(x_j)$ for $p < j \leq p+l$, and since $\alpha^{-1}(v') \in L_{v'+}(x_q)$ there exists a path from x_q to v' where the labels of all intermediate vertices in the path are smaller than both the labels of x_q and v'. Thus we have a path from x_p to v', where every intermediate vertex has a smaller label than x_p and v'. This is a contradiction since $\alpha^{-1}(v') \notin L_{v'+}(x_p)$. Now we return to our main proof. Since $L_{v'-}(x_q) \geq L_{v'-}(x_i)$ and $\alpha^{-1}(v') \notin L_{v'+}(x_i)$ for $0 \leq i < q$, while $\alpha^{-1}(v') \in L_{v'+}(x_q)$, we have $L_{v'+}(x_q) > L_{v'+}(x_i)$ for $0 \leq i < q$. It follows from Lemma 2.3 that $L_{z-}(x_q) > L_{z-}(u)$, where $0 < q \leq k$, since $L_{v'+}(x_q) > L_{v'+}(u = x_0)$. Now we have a contradiction since we assumed that $L_{z-}(x_i) \leq L_{z-}(u)$ where $0 < i < k$ and that $L_{z-}(u) \geq L_{z-}(v)$. ∎

The last case, where $|L_{z-}(u)| = |L_{z-}(v)|$ if and only if $L_{z-}(u) = L_{z-}(v)$ is now easy to prove. We can sum up the two previous lemmas as follows.

Lemma 3.3 *Assume that there is an unnumbered path $x_0, x_1, ..., x_k$ in G right before step $\alpha^{-1}(z)$ of Lex M, where $k \geq 1$, $u = x_0$, and $v = x_k$, and let $L_{z-}(x_i) \leq L_{z-}(u)$ where $0 < i < k$. Then we have*
1. $|L_{z-}(u)| > |L_{z-}(v)|$ if and only if $L_{z-}(u) > L_{z-}(v)$,
2. $|L_{z-}(u)| < |L_{z-}(v)|$ if and only if $L_{z-}(u) < L_{z-}(v)$,
3. $|L_{z-}(u)| = |L_{z-}(v)|$ if and only if $L_{z-}(u) = L_{z-}(v)$.

Proof. The first case is Lemma 3.1, while the second case is Lemma 3.2. The third case follows, since no alternatives are left. ∎

4 Lex M versus MCS-M

Lex M and MCS-M are not that different when it comes to altering labels and weights. If a vertex z is selected as the next vertex to be numbered for both algorithms, both Lex M and MCS-M do a search among unnumbered vertices that can be reached from z. In order to better compare the algorithms, these unnumbered vertices are partitioned into components.

Definition 4.1 *Let S be the set of numbered vertices, at some step of Lex M or MCS-M on $G = (V, E)$. Then an* unum component *is a connected component of $G(V \setminus S)$.*

Definition 4.2 *For any vertex u of G, CC_{u-} (resp. CC_{u+}) denotes the set of unum components of G right before (resp. after) numbering vertex u.*

In the proof that Lex M and MCS-M create exactly the same set of triangulations, we need some basic results regarding Lex M, MCS-M, and unum components. First we show that when Lex M or MCS-M processes a vertex in an unum component C they will only change the labels or weights of vertices contained in C. We then prove that if the length of the label in Lex M and the weight in MCS-M are the same for every vertex in an unum component C, then Lex M can choose a vertex z in C as the first vertex to be numbered in C if and only if MCS-M can choose z as the first vertex to be numbered in C. Then we prove that, under the same conditions, the length and the weight are still equal when a vertex in C is processed and the weight for MCS-M and labels for Lex M are updated.

Lemma 4.3 *In any execution of Lex M or MCS-M on a graph G, processing a vertex z of G only affects the unum component of CC_{z-} containing z (i.e. any other unum component of CC_{z-} is still an unum component of CC_{z+} with the same labels or weights).*

Proof. Let C be an unum component of CC_{z-} not containing z. It is evident that after removal of z, C is still an unum component of CC_{z+}. No labels or weights are changed in C, since for any vertex v whose label or weight is modified when processing z, there is a path of unnumbered vertices between z and v, so that v is in the same unum component of CC_{z-} as z. ∎

Lemma 4.4 *We consider two executions of Lex M and MCS-M respectively on a graph G. Let u and u' be vertices of G, and let C be a set of vertices of G such that C is an unum component of G right before processing u (resp. u') in the execution of Lex M (resp. MCS-M) and for every vertex v of C, $|L_{u-}(v)| = w_{u'-}(v)$. Then $hL_{u-}(C) = hW_{u'-}(C)$.*

Proof. We want to show that $hW_{u'-}(C) \subseteq hL_{u-}(C)$ and $hL_{u-}(C) \subseteq hW_{u'-}(C)$ and thus $hW_{u'-}(C) = hL_{u-}(C)$. The first step is to prove that $hW_{u'-}(C) \subseteq hL_{u-}(C)$. Let us on the contrary assume that there exists a vertex $m \in hW_{u'-}(C) \setminus hL_{u-}(C)$, and let l be any vertex in $hL_{u-}(C)$. The unum component C is connected, and every vertex in C is unnumbered. Thus there exists an unnumbered path $x_0, x_1, ..., x_k$ for $0 < k \leq |C| - 1$, where $l = x_0, m = x_k$, and $x_i \in C$ for $0 \leq i \leq k$. Then $L_{u-}(x_i) \leq L_{u-}(l)$ for $0 < i \leq k$ since $l \in hL_{u-}(C)$. We have $L_{u-}(l) > L_{u-}(m)$ since $l \in hL_{u-}(C)$ and $m \notin hL_{u-}(C)$. We have $w_{u'-}(l) \leq w_{u'-}(m)$ since $m \in hW_{u'-}(C)$. From the premises of the lemma, we then have that $|L_{u-}(l)| = w_{u'-}(l) \leq w_{u'-}(m) = |L_{u-}(m)|$. It follows that the path $x_0, x_1, ..., x_k$ is a contradiction to Lemma 3.3.

Next we want to prove that $hL_{u-}(C) \subseteq hW_{u'-}(C)$, and thus $hL_{u-}(C) = hW_{u'-}(C)$. Let us on the contrary assume that there exists a vertex $l \in hL_{u-}(C) \setminus hW_{u'-}(C)$, and let m be any vertex in $hW_{u'-}(C)$. Then there exists a path $x_0, x_1, ..., x_k$ for $0 < k \leq |C| - 1$, where $l = x_0, m = x_k$, and $x_i \in C$ for $0 \leq i \leq k$. We have $L_{u-}(x_i) \leq L_{u-}(l)$ for $0 < i \leq k$ since $l \in hL_{u-}(C)$. We have $w_{u'-}(l) < w_{u'-}(m)$ since $l \notin hW_{u'-}(C)$ and $m \in hW_{u'-}(C)$. Therefore the path $x_0, x_1, ..., x_k$ is a contradiction to Lemma 3.3. ∎

Lemma 4.5 *We consider two executions of Lex M and MCS-M respectively on a graph G. Let z be a vertex of G, and let C be a set of vertices of G such that C is an unum component of G right before processing z in both executions and for every vertex u of C, $|L_{z-}(u)| = w_{z-}(u)$. Then $|L_{z+}(u)| = w_{z+}(u)$ for every vertex u of $C \setminus \{z\}$.*

Proof. Let us on the contrary assume that $|L_{z+}(u)| \neq w_{z+}(u)$ for some $u \in C \setminus \{z\}$. From Lemma 4.3 we know that $z \in C$ if $|L_{z+}(u)| \neq w_{z+}(u)$. Two cases are possible. The first case is $|L_{z+}(u)| = w_{z+}(u) + 1$. There exists at least one path $x_0, x_1, ..., x_k$ for $k \geq 1$, where $u = x_0$, $z = x_k$, $x_i \in C$, and $L_{z-}(x_i) < L_{z-}(u)$ for $0 < i < k$, since $|L_{z+}(u)| = |L_{z-}(u)| + 1$ and C is an unum component of G containing u right before processing z. Then for every such path there exists a vertex x_j where $0 < j < k$ such that $w_{z-}(x_j) \geq w_{z-}(u)$, since $w_{z+}(u) = w_{z-}(u)$. The path $x_0, x_1, ..., x_j$ is a contradiction to Lemma 3.3 because (1) $L_{z-}(x_i) < L_{z-}(u = x_0)$ for $0 < i \leq j$, and specifically $L_{z-}(u) > L_{z-}(x_j)$, and (2) $w_{z-}(u) \leq w_{z-}(x_j)$ and hence, due to our assumption, $|L_{z-}(u)| \leq |L_{z-}(x_j)|$. The second case is when $|L_{z+}(u)|+1 = w_{z+}(u)$ for some vertex $u \in C \setminus \{z\}$. Then there has to exist at least one path $x_0, x_1, ..., x_k$ for some $k \geq 1$, where $u = x_0$, $z = x_k$, $x_i \in C$ for $0 \leq i \leq k$, and $w_{z-}(x_i) < w_{z-}(u) \leq w_{z-}(z)$ for $0 < i < k$, since $w_{z+}(u) = w_{z-}(u) + 1$. Then for every such path there exists a vertex x_j for $0 < j < k$ such that $L_{z-}(x_j) \geq L_{z-}(u)$, since $|L_{z+}(u)| = |L_{z-}(u)|$. Let j be the smallest number such that $L_{z-}(x_j) \geq L_{z-}(u)$. The path $x_0, x_1, ..., x_j$ is a contradiction to Lemma 3.3 because (1) $L_{z-}(x_i) < L_{z-}(u = x_0)$ for $0 < i < j$ and moreover $L_{z-}(u) \leq L_{z-}(x_j)$, and (2) $w_{z-}(u) > w_{z-}(x_j)$ and hence, due to our assumption, $|L_{z-}(u)| > |L_{z-}(x_j)|$. ∎

The three previous lemmas are local observations, and require that Lex-M and MCS-M have an unum component consisting of the same vertices, where the weight in MCS-M is equal to the length of the label in Lex M for every vertex in the unum component. The following definition will be useful to formalize the fact that both algorithms break ties in the same way in unum components.

Definition 4.6 *Let $G = (V, E)$ and ϕ be a mapping from the set of all subsets of V to V, such that if $\phi(S) = u$ then $u \in S$, for each $S \subseteq V$. An execution of Lex M (resp. MCS-M) on G is said to be* compatible with ϕ *if for any vertex u of G, $u = \phi(hL_{u-}(C))$ (resp. $\phi(hW_{u-}(C))$), where C is the unum component of CC_{u-} containing u.*

The idea behind ϕ is the following. If S is a set of vertices in Lex M with the highest label belonging to an unum component, or a set of vertices in MCS-M with the highest weight belonging to an unum component, then $\phi(S)$ is the vertex that is chosen next among vertices of this unum component.

Note that two different executions of Lex M (resp. MCS-M) on G can be compatible with the same mapping ϕ, since ϕ tells which vertex to choose next to be numbered in a given unum component, but does not tell in which unum component to choose the next vertex to be numbered in case some vertices with largest label or weight lie in different unum components.

Lemma 4.7 *We consider two executions of Lex M and MCS-M respectively on a graph $G = (V, E)$. If these executions are compatible with the same mapping ϕ from the set of all subsets of V to V, then they produce the same minimal triangulation of G.*

Proof. We define the following property $P(k)$.
$P(k)$: for any vertices u and u' of G and any set C of k vertices of G, if C is

an unum component of G right before processing u (resp. u') in the execution of Lex M (resp. MCS-M) and for every vertex v of C, $|L_{u-}(v)| = w_{u'-}(v)$ then, the fill edges produced when processing the vertices of C are the same in both executions.
It is sufficient to prove that $P(k)$ holds for $k = n$, since in that case $C = V$, which is an unum component at the beginning of both executions with empty labels and null weights, hence the sets of fill edges produced are the same in both executions.
Let us prove that $P(k)$ holds for k from 1 to n by induction on k.
$P(1)$ is true since the unique vertex of C can produce no fill edge by Lemma 4.3. We assume that $P(k)$ holds. Let us show that $P(k+1)$ holds. Let u and u' be vertices of G, and let C be a set of $k+1$ vertices of G such that C is an unum component of G right before processing u (resp. u') in the execution of Lex M (resp. MCS-M) and for every vertex v of C, $|L_{u-}(v)| = w_{u'-}(v)$. By Lemma 4.3, these conditions are maintained until a vertex z (resp. z') of C is numbered for the first time, from the moment when u (resp. u') is about to be numbered in the execution of Lex M (resp. MCS-M) (possibly $z = u$ or $z' = u'$, if u or u' belongs to C). By Lemma 4.4, $hL_{z-}(C) = hW_{z'-}(C)$, and as both executions are compatible with ϕ, $z = \phi(hL_{z-}(C)) = \phi(hW_{z'-}(C)) = z'$. By Lemma 4.5, $|L_{z+}(v)| = w_{z+}(v)$ for every vertex v of $C \setminus \{z\}$. So the processing of z modifies the labels or weights of the same vertices of C in both executions, and since by Lemma 4.3 the labels or weights of the vertices of $G \setminus C$ are unchanged, the processing of z produces the same fill edges in both executions. Moreover, the new unum components obtained from C by removing z are the same in both executions. So, by the induction hypothesis on these new unum components which contain at most k vertices and for which the condition on labels and weights holds after processing z, we have the fill edges produced when processing the vertices of $C \setminus \{z\}$ are the same in both executions, which completes the proof. ∎

In order to complete the proof that Lex M and MCS-M produce the same set of chordal graphs, two more arguments are required. The first is to show that for any execution of Lex M (resp. MCS-M) there exists a mapping ϕ compatible with this execution. The second is to show that for any mapping ϕ from the set of all subsets of V to V such that for any subset S of V, $\phi(S)$ belongs to S, there is an execution of MCS-M (resp. Lex M) compatible with ϕ. Then the rest will follow from Lemma 4.7.

Theorem 4.8 *Lex M and MCS-M produce the same minimal triangulations of a given graph $G = (V, E)$.*

Proof. Observe that for any execution of Lex M on G producing the triangulated graph H there exists a compatible mapping ϕ. This mapping ϕ can simply be constructed as follows: For every vertex $z \in V$ set $\phi(hL_{z-}(C))$ to z, where C is the unum component in CC_{z-} containing z. Any mapping ϕ which fulfills this requirement will be compatible with the execution of Lex M producing H. Note that during Lex M, $hL_{z-}(C) \neq hL_{z'-}(C')$ for all vertices $z \neq z'$ with $z \in C$ and $z' \in C'$ where $C \in CC_{z-}$ and $C' \in CC_{z'-}$, since the highest numbered of z and z' does not belong to both sets. Thus we never consult $\phi(S)$ for the same set S of vertices more than once.

We now consider an execution of MCS-M on G compatible with ϕ. Such an execution exists. At each step it is sufficient to choose an unum component C containing a vertex with largest weight and to choose $\phi(hW(C))$ as next vertex to be numbered. By Lemma 4.7, this execution of MCS-M produces the graph H.

The proof in the other direction is completely symmetric. ∎

5 Conclusion

Even though MCS-M and Lex M can create different orderings, we prove that they create the same set of triangulations, and thereby answer an open question given in [3]. We show this by defining unum components, which are the connected subgraphs when the numbered vertices are removed from the graph. Then we show that two executions of Lex M and MCS-M breaking ties in the same way in unum components compute the same minimal triangulation of the input graph, so that Lex M and MCS-M compute the same set of minimal triangulations of any graph.

We also observe that each of the unum components can be computed individually since they do not affect each other. This property could possibly be used to improve the practical running time for both algorithms.

Acknowledgment:

The author wishes to thank Pinar Heggernes for her useful comments and suggestions, and the two anonymous referees who assisted in the presentation of this paper.

References

[1] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Alg. Disc. Meth.*, 8:277–284, 1987.

[2] A. Berry. A wide-range efficient algorithm for minimal triangulation. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'99)*, pages S860–S861, 1999.

[3] A. Berry, J. R. S. Blair, P. Heggernes, and B. W. Peyton. Maximum cardinality search for computing minimal triangulations of graphs. *Algorithmica*, 39:287–298, 2004.

[4] A. Berry, J.P. Bordat, P. Heggernes, G. Simonet, and Y. Villanger. A wide-range algorithm for minimal triangulation from an arbitrary ordering. *Journal of Algorithms*, 58(1):33–66, 2006.

[5] A. Berry, P. Heggernes, and Y. Villanger. A vertex incremental approach for dynamically maintaining chordal graphs. In *Proceedings 14th International Symposium on Algorithms and Computation - ISAAC 2003*, pages 47 – 57. Springer Verlag, 2003. LNCS 2906.

[6] J. R. S. Blair, P. Heggernes, and J. A. Telle. A practical algorithm for making filled graphs minimal. *Theor. Comput. Sci.*, 250:125–141, 2001.

[7] E. Dahlhaus. Minimal elimination ordering inside a given chordal graph. In R. H. Möhring, editor, *Graph Theoretical Concepts in Computer Science - WG '97*, pages 132–143. Springer Verlag, 1997. Lecture Notes in Computer Science 1335.

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

[9] T. Ohtsuki. A fast algorithm for finding an optimal ordering in the vertex elimination on a graph. *SICOMP*, 5:133–145, 1976.

[10] S. Parter. The use of linear graphs in Gauss elimination. *SIAM Review*, 3:119–130, 1961.

[11] B. W. Peyton. Minimal orderings revisited. *SIAM J. Matrix Anal. Appl.*, 23(1):271–294, 2001.

[12] D. Rose, R.E. Tarjan, and G. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comput.*, 5:146–160, 1976.

[13] M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM J. Alg. Disc. Meth.*, 2:77–79, 1981.