Recognizing and computing minimal triangulations efficiently

Daniel Meister

Report No. 302 Dezember 2002

Preprint-Reihe
Fakultät für Mathematik und Informatik
Universität Würzburg

Recognizing and computing minimal triangulations efficiently

Daniel Meister
Theoretische Informatik
Bayerische Julius-Maximilians-Universität Würzburg
97074 Würzburg, Germany
meister@informatik.uni-wuerzburg.de

Abstract. A graph H is a minimal triangulation of a graph G if H is a chordal supergraph of G on the vertex set of G and no proper subgraph of H is a chordal supergraph of G. We introduce a partition refinement algorithm `min-LexBFS` based on `LexBFS` and show that it yields linear-time algorithms to compute minimal triangulations of co-comparability graphs and AT-free claw-free graphs. These triangulation algorithms are used to improve approximation algorithms for the bandwidth of co-comparability and AT-free claw-free graphs. We present a certifying recognition algorithm for proper interval graphs that uses our refinement strategy.

1. Introduction. A triangulation H of a given graph G is a spanning supergraph of G that is chordal, i.e., every cycle in H of length at least 4 contains a chord. Therefore, triangulating G means adding additional edges to G. If there is no triangulation of G that is properly contained in H, we call H a minimal triangulation of G; H is *minimal* with regard to the set of edges. Minimal triangulations of arbitrary graphs can be computed in time $\mathcal{O}(nm)$ [42]. If we are interested in the smallest number of additional edges to make G chordal, we ask for a *minimum* triangulation; the related decision problem MINIMUM FILL-IN, given a graph G and a natural k, is there a triangulation of G that has at most k additional edges, is NP-complete even for co-bipartite graphs [44].

Minimal triangulations first arised in the context of vertex eliminiation schemes. Given a sparse symmetric positive definite matrix that represents the coefficients of a system of linear equations, we can associate a graph by assigning a vertex to every row (and column simultaneously) of the matrix and making two vertices adjacent if and only if their corresponding row and column define a non-zero element in the matrix. Rose [41] showed that the non-zero elements in the matrix produced by Gaussian elimination correspond exactly to the edges that are added to the graph when vertices are eliminated in the same ordering as rows are during the Gaussian elimination by making their neighbourhoods a clique. If no edge is created, which means that no new non-zero element is generated, the process is called a perfect elimination scheme, and the graph must be chordal. Every chordal graph has a perfect elimination scheme [16]. Although it is desirable to eliminate vertices or rows in order to add the smallest number of new edges (or non-zero elements, respectively), we now know that it is hard to compute such an optimal elimination scheme, which has been proved by Yannakakis in 1981 [44]. In 1976, Rose, Tarjan, and Lueker presented a polynomial-time algorithm to compute a minimal triangulation of a graph. They already conjectured the NP-completeness of MINIMUM FILL-IN [42].

During the last decade, other algorithms for computing minimal triangulations were presented by Kloks, Kratsch, and Müller [26] and Berry [1]. The most recent, due to Berry, Blair, and Heggernes, was presented at WG 2002 [2]. All mentioned algorithms are suitable for arbitrary graphs and run in time $\mathcal{O}(n^3)$, even $\mathcal{O}(nm)$ except for [26] (see also [22]). Of course, two questions immediately arise: Is $\mathcal{O}(nm)$ time best possible in case of arbitrary graphs, and can we do better by restricting the input graphs? In this paper, we give an answer to the second question. In fact, we will show that for AT-free claw-free and co-comparability graphs minimal triangulations can be computed in linear time, even though both classes contain the class of co-bipartite graphs.

Besides the Gaussian elimination process there is another aspect which makes minimal triangulation be of a certain interest. A minimal triangulation of a graph G can be regarded as a *minimal embedding* of G into a chordal graph, which means to find an inclusion-minimal set F of edges to make G chordal. It is obvious that G is chordal if and only if F is empty. So, a minimal triangulation algorithm is also a recognition algorithm for chordal graphs. Of course, in the general case the worst case time complexity of all known triangulation algorithms does not make them a good choice for recognizing chordal graphs. However, in case of interval graphs the situation changes since they are also co-comparability graph. So, we can use our linear-time triangulation algorithm for co-comparability graphs to recognize interval graphs.

Rose, Tarjan, and Lueker introduced a linear-time algorithm to compute a perfect elimination scheme for a chordal graph that is most often refered to as `LexBFS` [42], which is short for *lexicographic breath first search*. Basically, it is some kind of breath first search, and ties are broken with respect to already visited vertices. `LexBFS` may be regarded as a partition refinement algorithm [21]. Partition refinement seems to be a strong algorithmic paradigm in algorithm design [36]. We use a similar approach and introduce a new algorithm `min-LexBFS` which is in fact related to `LexBFS` and based on partiton refinement. Both algorithms generate orderings of the vertex set of the input graph with special properties. Although `LexBFS` and `min-LexBFS` generally differ on an input graph, which means that both algorithms output different orderings, they have some important properties in common. In [3], Berry and Bordat introduced the notion of a moplex, which is a maximal clique-module whose neighbourhood is a minimal separator, and they showed that `LexBFS` alsways numbers the vertices of a moplex last. We will show that the same is true for `min-LexBFS`. But we will also show that there are graphs for which the set of moplexes numbered last by `LexBFS` is disjoint to that of `min-LexBFS`. As an application of this algorithm, we will obtain a simple and linear-time algorithm that computes a minimal triangulation of an AT-free claw-free graph. Moreover, it turns out that `min-LexBFS` perfectly characterizes the minimal triangulations of an AT-free claw-free graph, which means that H is a minimal triangulation of an AT-free claw-free graph G if and only if H can be the output of a double-sweep `min-LexBFS` run on G. In case of co-comparability graphs, the situation is different. We also present a linear-time algorithm computing a minimal triangulation of a co-comparability graph, but there is a co-comparability graph and a minimal triangulation of it that cannot be generated by our algorithm.

Another part of this paper is dedicated to a new recognition algorithm for proper interval graphs. Many such algorithms are known, the most recent of which is due to Corneil and applies a simple 3-sweep `LexBFS` to the input graph [6]. The output is a proper interval ordering for the graph if and only if it is a proper interval graph. Our algorithm will be quite similar, a 3-sweep `min-LexBFS`, but the new feature is that it provides evidence in the rejecting case. This means that our algorithm *convinces* the user of its decision by means of easily verifiable certificates.

The paper is organized as follows. In the next section, we give basic notions and prove some useful results concerning interval graphs. In section 3, we formally introduce `min-LexBFS` and give an easy and useful characterization of the orderings that are generated by the algorithm. As an immediate corollary, we conclude that interval orderings, defined in section 2, can be generated by `min-LexBFS`. In section 4, we show that `min-LexBFS` always numbers last all vertices of a moplex, which in itself is a very interesting property of `min-LexBFS`. Section 5 is dedicated to the linear-time certifying recognition algorithm for proper interval graphs. In sections 6 and 7, we consider the computation of minimal triangulations of AT-free claw-free graphs and co-comparability graphs, respectively. Also in section 7, we give an approximation algorithms for the bandwidth of co-comparability and AT-free claw-free graphs with worst case performance ratio 2, which improve known results.

2. Preliminaries. We consider only simple, finite, undirected graphs $G = (V, E)$, where V or $V(G)$ denotes the *vertex set* of G and E or $E(G)$ its *edge set*; edges of G may be denoted by uv or $\{u, v\}$, which means that u and v are *adjacent*. We denote by n the number of vertices and by m the number of edges in G. Further definitions and denotations follow mainly classical books like [19], especially in case of notions not explicitly defined here. The exact representation of a graph is especially of algorithmic interest. We assume all graphs to be given by their adjacency lists. The *(open) neighbourhood* $N_G(u)$, or simply $N(u)$ if the context is clear, of a vertex $u \in V$ is the set of all vertices of G adjacent to u; $N_G[u] = N_G(u) \cup \{u\}$ denotes the *closed* neighbourhood of u. For $S \subseteq V$, $N_G(S) = \bigcup_{u \in S} N_G(u) \setminus S$ denotes the open neighbourhood of S in G. By $G[S]$, we denote the subgraph of G induced by S; $G \setminus S = G[V \setminus S]$. For some vertex $u \in V$ and an induced subgraph G' of G that does not contain u as vertex, $G'+u$ is the subgraph of G induced by $V(G') \cup \{u\}$. The graph $G-e$ for some edge e in G is obtained from G by deleting e in G. A *path* P of length $k-1$ in a graph is a sequence $(u_1, \ldots, u_k)$ of k different vertices with edges between consecutively occuring vertices; P is a u_1, u_k-path. A *cycle* $(u_1, \ldots, u_k)$ of length k is a path with u_1 and u_k adjacent. A *chord* in such a cycle is an edge between two non-consecutive vertices. A chord is a *unique chord* in a cycle if there is no other edge in G that forms a chord in this cycle. A cycle is called *chordless* if there is no edge in G that forms a chord in this cycle. A chordless cycle must have length at least 4. A graph is *chordal* if it does not contain a chordless cycle. Every chordal graph contains a *simplical* vertex [14], i.e., a vertex whose neighbourhood is a clique. An *interval graph* is the intersection graph of a family of closed intervals of the real line; we will call such a family an *interval model* I, and the corresponding interval graph will be denoted by $G(I)$. Interval graphs are chordal. Three pairwise non-adjacent vertices u, v, w form an *asteroidal triple* (*AT* for short) if there is a path between every two of them avoiding the neighbourhood of the third. A graph is *AT-free* if it does not contain any AT. AT-free graphs have been studied extensively in [8] and [10].

Theorem 1. **[30]** *A chordal graph is an interval graph if and only if it is AT-free.*

For a graph $G = (V, E)$, a *triangulation of* G is a chordal supergraph $H = (V, E \cup F)$, $E \cap F = \emptyset$, of G. H is a *minimal* triangulation of G if there is no triangulation H' of G such that $G \subseteq H' \subset H$; a minimal triangulation of a graph is minimal with respect to edge set inclusion. If G is not connected H is a minimal triangulation of G if and only if there is a 1-1-correspondence between the components of G and H and every component of H is a minimal triangulation of the corresponding component of G. Rose, Tarjan, and Lueker studied minimal triangulations and gave two useful characterizations.

Theorem 2. **[42]** *Let $G = (V, E)$ and $H = (V, E \cup F)$, $E \cap F = \emptyset$, be two graphs. The following two statements are equivalent to H being a minimal triangulation of G:*

i. *For every $e \in F$, e is unique chord in a cycle of length 4 in H,*

ii. *For every $e \in F$, the graph $H - e$ is not chordal.*

A vertex of a graph is *universal* if it is adjacent to every other vertex. A *claw* is a graph on four vertices containing three pairwise non-adjacent vertices and a universal one. A graph is *claw-free* if it does not contain a claw, i.e., no four vertices induce a claw. *AT-free claw-free graphs* are AT-free graphs that are claw-free. A *proper interval graph* is an interval graph for which there is an interval model such that no interval properly contains another; a *unit interval graph* is an interval graph for which there is an interval model such that all intervals are of the same length. The notions of chordal AT-free claw-free graphs, proper interval graphs, and unit interval graphs are equivalent [40]. Surprisingly, minimal triangulations are suitable to characterize AT-free graphs and AT-free claw-free graphs.

Theorem 3. **[34], [39]** *Let G be a graph.*

i. G is AT-free if and only if all its minimal triangulations are AT-free.

ii. G is AT-free claw-free if and only if all its minimal triangulations are AT-free claw-free.

In fact, a graph is AT-free if and only if each of its minimal triangulations is an interval graph, and a graph is AT-free claw-free if and only if all its minimal triangulations are proper interval graphs. Therefore, minimal triangulations of AT-free graphs can be represented by an appropriate interval model. As far as we know there are not many other graph classes that can be characterized in terms of their minimal triangulations.

Graphs and orderings. Some of the previously defined graph classes provide an alternative definition using vertex orderings. Let $G = (V, E)$ be a graph, and let $x_1, \ldots, x_n$ be the vertices of G. For an ordering $\sigma = \langle x_1, \ldots, x_n \rangle$ on V, $\prec_\sigma$ is defined as follows: for any two vertices $u, v \in V$, $u \prec_\sigma v$ if and only if $u = x_i$ and $v = x_j$, for suitable i and j, and $i < j$. Equivalently, we say that u is *to the left* of v, v is *to the right* of u, with respect to σ. By σ^{-1}, we denote the *inverse* of σ, which means $\sigma^{-1}(x_i) = i$; $\vec{\sigma} = \langle x_n, \ldots, x_1 \rangle$ denotes the *reverse* of σ. The *left (open) neighbourhood* of u with regard to σ is denoted by $N_G^-(u)$ and contains all neighbours of u that are to the left of u with regard to σ. Since the context will always be clear, we avoid a sub- or superscript σ. The right neighbourhood $N_G^+(u)$ and the closed neighbourhoods $N_G^-[u]$ and $N_G^+[u]$ of u are defined similarly. By $\mathrm{lm}_{G,\sigma}(u)$, we denote the leftmost vertex in $N_G^-[u]$ (with respect to σ); subscripts may be omitted if it is possible. The ordering σ is a *perfect elimination scheme* for G if x_i is a simplicial vertex in $G[\{x_i, \ldots, x_n\}]$ for all i, $1 \leq i \leq n$.

Theorem 4. **[16]** *A graph is chordal if and only if it has a perfect elimination scheme.*

It is easy to see that σ is a perfect elimination scheme for G if and only if for every three vertices $u, v, w \in V$, $u \prec_\sigma v \prec_\sigma w$, $uv \in E$ and $uw \in E$ implies $vw \in E$ [42]. By $H = (G, \sigma)$, we denote the *filled-in* graph with respect to G and σ. H is that graph such that the following holds. If G has only one vertex, $H = G$. If $n > 1$, let $u \in V$ be the first vertex in σ; $N_H(u) = N_G(u)$. Let G' be the subgraph of G obtained by making u simplicial and removing it from G; let σ' be obtained from σ by removing u. $H = H' + u$, where $H' = (G', \sigma')$. By theorem 4, H is chordal, and σ is a perfect elimination scheme for H. H is a triangulation of G. If H is a minimal triangulation of G, σ is a *minimal elimintation scheme* for G.

The graph G is *transitively orientable* if each edge can be assigned an orientation such that for all three vertices $u, v, w \in V$, $uv \in F$ and $vw \in F$ implies $uw \in F$ where F denotes the set of oriented edges of G. If $uv \in F$, then the orientation is from u to v. If G provides a transitive orientation of its edges it is a *comparability graph*; complements of such graphs are *co-comparability graphs*. They can be defined alternatively via vertex orderings. G is a co-comparability graph if and only if there is a *co-comparability ordering* σ of V such that for all three vertices $u, v, w \in V$, $u \prec_\sigma v \prec_\sigma w$, $uw \in E$ implies $uv \in E$ or $vw \in E$ [29]. Co-comparability graphs are AT-free [20].

The graph G is an interval graph if and only if there is an *interval ordering* σ of the vertices of G such that for every three vertices $u, v, w \in V$, $u \prec_\sigma v \prec_\sigma w$, $uw \in E$ implies $vw \in E$ [35]. Obviously, σ is both a perfect elimination scheme for G and a co-comparability ordering. If σ is both a minimal elimination scheme for G and an interval ordering for $H = (G, \sigma)$, we call it a *minimal interval elimination scheme* for G. In other words, σ is a minimal interval elimination scheme if each edge in $E(H) \setminus E$, that is required to make σ an interval ordering, is unique chord in a cycle of length 4 in H. Since we only regard AT-free graphs, minimal interval elimination schemes will play a central role in our investigations.

Lemma 5. *Let $G = (V, E)$ be a graph, and let σ be a vertex ordering for G; let $H = (G, \sigma)$. Then, for every vertex $u \in V$, $\mathrm{lm}_{H,\sigma}(u) = \mathrm{lm}_{G,\sigma}(u)$. Furthermore, if H is an interval graph and σ an interval ordering for H, an interval model of H can be computed in time linear in G.*

Proof: Suppose there is a vertex $u \in V$ such that $\mathrm{lm}_H(u) \prec_\sigma \mathrm{lm}_G(u)$, $v = \mathrm{lm}_H(u)$. By definition of H, there must be a neighbour of u in H that is a neighbour of v and to the left of v with respect to σ, which contradicts the definition of lm. Hence, $uv \in E$. To construct the model I, every vertex u of H is represented by an interval defined by left and right endpoints $\ell(u)$ and $r(u)$, respectively. For all vertices $u \in V$, let $r(u) = \sigma^{-1}(u)$. Let $v = \mathrm{lm}_G(u)$. If $v \prec_\sigma u$, let $\ell(u) = \sigma^{-1}(v) - \frac{1}{4}$; otherwise, let $\ell(u) = \sigma^{-1}(u) - \frac{1}{2}$. Obviously, this algorithm runs in time linear in G. It is also clear that $H \subseteq G(I)$. Let u and v be two vertices, $u \prec_\sigma v$, and their corresponding intervals intersect (i.e., $\ell(v) < r(u)$). By construction, $\mathrm{lm}(v) \preceq_\sigma u$, and since σ is an interval ordering for H, $uv \in E(H)$. ∎

The proof also shows that the characterizations of interval graphs in terms of interval models and interval orderings are very closely related. Given an interval model, to obtain an interval ordering it is sufficient to order the vertices according to the right endpoints of their corresponding intervals [35]. The following statement shows that it is easy to verify whether a given ordering is an interval ordering for a given graph.

Corollary 6. *Let $G = (V, E)$ be a graph. The vertex ordering $\sigma = \langle x_1, \ldots, x_n \rangle$ for G is an interval ordering for G if and only if*

$$m = \sum_{1 \le i \le n} i - \sigma^{-1}(\mathrm{lm}(x_i)).$$

Proof: Since for each edge $uv \in E$, either $u \in N^-(v)$ or $v \in N^-(u)$, and $i - \sigma^{-1}(\mathrm{lm}(x_i)) \geq |N^-(x_i)|$,

$$m = \sum_{1 \le i \le n} |N^-(x_i)| \le \sum_{1 \le i \le n} i - \sigma^{-1}(\mathrm{lm}(x_i))$$

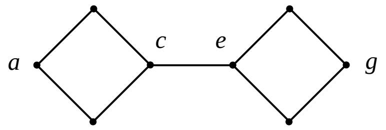

Figure 1. A small counterexample.

for every graph. If σ is an interval ordering, $i - \sigma^{-1}(\mathrm{lm}(x_i)) = |N^-(x_i)|$ for every i, $1 \leq i \leq n$. If σ is not an interval ordering, there is some vertex $x_i \in V$ such that $i - \sigma^{-1}(\mathrm{lm}(x_i)) > |N^-(x_i)|$. ∎

Finally, σ is a *proper interval ordering* of the vertices of G if for every three vertices $u, v, w \in V$, $u \prec_\sigma v \prec_\sigma w$, $uw \in E$ implies both $uv \in E$ and $vw \in E$. A graph is a proper interval graph if and only if its vertices admit a proper interval ordering [31]. We observe that σ is a proper interval ordering if and only if σ and $\vec{\sigma}$ are interval orderings. In consequence of corollary 6, we can easily verify whether a given vertex ordering for G is a proper interval ordering for G.

Distances and separators. Let $G = (V, E)$ be a graph. For two verties u and v, a *shortest* u, v-path is a u, v-path that has shortest length among all u, v-paths. The length of a shortest u, v-path is the *distance* of u and v, denoted by $d_G(u, v)$. If there is no u, v-path, $d_G(u, v) = \infty$. For G connected and a vertex $z \in V$, define the *BFS-levels* $S_0, \ldots, S_k$ of G with root z such that $u \in S_i$, $0 \leq i \leq k$, if and only if $d_G(z, u) = i$; k is the largest number such that there is a vertex in distance k to z. An ordering σ is a *BFS-ordering* for G if for every two vertices $u, v \in V$, $u \in S_i$, $v \in S_j$, $u \prec_\sigma v$ implies $i \geq j$, where $S_0, \ldots, S_k$ are the BFS-levels of G with root $\sigma(n)$. A BFS-ordering σ for G with root z is a BFS-ordering for G' if the BFS-levels of G and G' with root z are equal. The *diameter* $\mathrm{diam}(G)$ of G is equal to the longest distance between two vertices of G; if G is not connected, $\mathrm{diam}(G) = \infty$.

Lemma 7. *Let $G = (V, E)$ be a connected graph, and let σ be a BFS-ordering for G with root $z = \sigma(n)$. Then, σ is a BFS-ordering for $H = (G, \sigma)$.*

Proof: Let $S_0, \ldots, S_k$ be the BFS-levels of G with root z, and let $S'_0, \ldots, S'_\ell$ be the BFS-levels of H with root z. Suppose there is i such that $S_i \neq S'_i$; let i be smallest possible. Then, $S_i \subset S'_i$, and there is a vertex $u \in S'_i \setminus S_i$. By definition of H, there must be vertices $v \in S_{i-1}$ and x, $x \prec_\sigma u$, such that x is a common neighbour of u and v in G. Then, $x \in S_i$, and by $x \prec_\sigma u$, $u \in S_i$, too, which contradicts the assumption. ∎

We immediately observe that the converse of the lemma is not true in the general case. Consider graph G depicted in figure 1. Obviously, $H = G \cup \{ac, eg\}$ is a minimal triangulation of G, and no BFS-ordering for H is a BFS-ordering of G. We conclude that there is no BFS-ordering σ for G such that $H = (G, \sigma)$.

For $S \subseteq V$, $a, b \in V$, S is an *ab-separator* in G if $G \setminus S$ does not contain an a, b-path; S is a *minimal* ab-separator if no proper subset of S is an ab-separator. A *minimal separator* in G is a minimal ab-separator for some vertices $a, b \in V$. A graph G is chordal if and only if every minimal separator of G is a clique [14].

Lemma 8. **[19]** *Let $G = (V, E)$ be a graph, $S \subseteq V$. S is a minimal separator in G if and only if there are two different components in $G \setminus S$ such that every vertex in S has a neighbour in both components.*

3. The min-LexBFS algorithm. In their paper about algorithmic aspects of vertex elimination processes on graphs, Rose, Tarjan, and Lueker [42] gave two possible ways to understand `LexBFS`. They used a label for each vertex, and during a run on a given graph, numbers were assigned to labels. The vertex to be chosen next was one of largest label with respect to lexicographic order. For implementation aspects, they avoided the explicit computation of the lexicographic order by using chained lists. Paige and Tarjan [36] and Habib et. al. [21] used the partition refinement technique to efficiently solve a variety of problems, such as sorting a set of words lexicographically, recognizing interval graphs, and computing transitive orientations.

Let $C_1, \dots, C_k$ be non-empty sets. By putting them into a list $\langle C_1, \dots, C_k \rangle$, we define an order on them. We say that C_i is *smaller than* or *left of* C_j, denoted by $C_i \prec C_j$, if and only if $i < j$. An important operation on these sets, henceforth called *boxes*, is *partitioning*. Let C_i be some box. We partition it by some rule into C' and C'', define C' to be left of C'', call C' the *left partition box*, C'' the *right partition box*, and obtain the resulting list $\langle C_1, \dots, C_{i-1}, C', C'', C_{i+1}, \dots, C_k \rangle$. This operation will shortly be denoted by *partitioning* C_i *into* $\langle C', C'' \rangle$, or even $L : C_i \leftarrow \langle C', C'' \rangle$ for L the name of the current list. Each such list contains only non-empty boxes, which means that empty boxes are instantly removed. If L contains k boxes, the *rightmost* box is C_k.

Let $G = (V, E)$ be a graph. By a sequence of partition operations and a certain selection rule for vertices, we want to compute an ordering of the vertices of G. Starting with a list of only one box containing all vertices, in each step we chose a vertex u from the rightmost box of the maintained list and partition the leftmost box containing a neighbour of u. The vertex ordering is given by the ordering of the chosen vertices.

```
min-LexBFS(G) returns σ:
begin
    L := ⟨V⟩;
    for i := n downto 1 do
        pick vertex u from the rightmost box of L;
        σ(i) := u;
        let C be the leftmost box in L containing a neighbour of u;
        L : C ← ⟨C \ N_G(u) , C ∩ N_G(u)⟩
    end for
end.
```

After picking the last vertex of the rightmost box, this box is removed from the list. If, for some picked vertex u, there is no box containing a neighbour of u, the list remains unchanged. If all vertices in C are neighbours of u, the list also remains unchanged, but this is because of $C \setminus N(u) = \emptyset$. Of course, u has nevertheless been picked from the rightmost box. If C also contains non-neighbours of u, we say that u *separates* x and z for any $x \in C \setminus N(u)$ and $z \in C \cap N(u)$. If we want to start with a fixed vertex, we can let it be a second input parameter, or we initialize list L with a pre-partition. If, for example, we want s to be start vertex, we initialize $L = \langle V \setminus \{s\}, \{s\} \rangle$. We will call σ a *min-LexBFS-ordering* for G if σ can be the output of some run of `min-LexBFS` on G. If we partition **all** boxes each time instead of only the leftmost box containing a neighbour of the currently chosen vertex, we obtain `LexBFS` (see also [42] and [21]). A first observation we could draw from comparing both algorithms may be that the deletion of a still unprocessed vertex from the

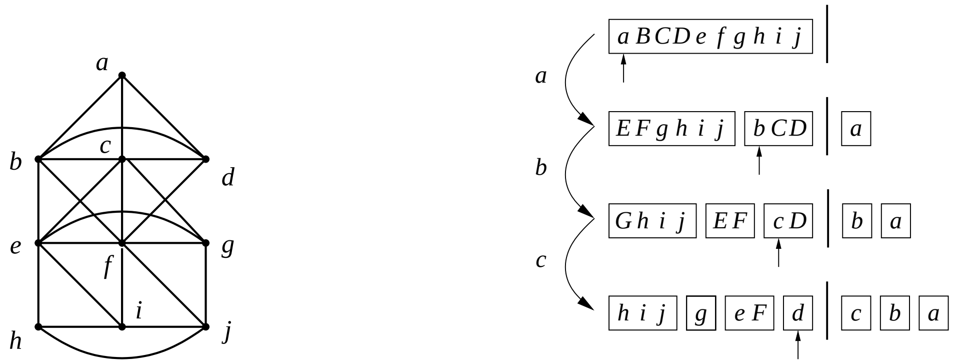

Figure 2. A graph and the beginning of a `min-LexBFS` run on it.

graph during a run does not effect the final ordering generated by `LexBFS`, but can cause errors for `min-LexBFS`. In other words, the deletion of a vertex from a LexBFS-ordering does not impede the subsequence to its right from being part of a LexBFS-ordering. Whereas, in case of a min-LexBFS-ordering, even the subsequence to the right of the deleted vertex may lose the property of being a min-LexBFS-ordering.

Fact 9. *Let $G = (V, E)$ be a connected graph, and let σ be a min-LexBFS-ordering for G. Then, σ is a BFS-ordering.*

Proof: Suppose σ is not a BFS-ordering; let $S_0, \ldots, S_k$ be the BFS-levels of G with root $\sigma(n)$, Then, there are two vertices u, v, $u \prec_\sigma v$, $u \in S_i$, $v \in S_j$, $i < j$. Without loss of generality, we assume u to be rightmost among all those vertices fulfilling this property. When the rightmost neighbour of u with respect to σ is chosen, u and v are both contained in the leftmost box. The subsequent partition results in u being in the right, v being in the left partition box, which contradicts the assumption. ∎

Figure 2 shows a graph and the first iteration steps of a `min-LexBFS` run on it beginning with a. At each step, the upright arrow points to the currently chosen vertex, and the capital letters represent its yet unselected neighbours. To the right of the upright (separation) line, the final sequence is iteratively generated. Both the lists to the left and to the right of the separation line are ordered from left to right. The next chosen vertex will be f. The resulting vertex ordering will be $\langle h, i, j, g, e, f, d, c, b, a\rangle$. In fact, every min-LexBFS-ordering of the pictured graph that ends with a starts with the sequence $\langle h, i, j\rangle$. Now, we run `LexBFS` on the same graph and obtain, for instance, $\langle h, j, i, g, e, f, d, c, b, a\rangle$, and by trying all possibilities, we observe that every LexBFS-ordering that ends with a starts with $\langle h, j, i\rangle$. We conclude that there are graphs for which there are min-LexBFS-orderings that are not LexBFS-ordering, and vice versa. More general, `min-LexBFS` does not always generate LexBFS-orderings.

Proposition 10. *The algorithm* `min-LexBFS` *can be implemented to run in linear time.*

Proof: Generally, we use the same implementation as it was presented in [42] for `LexBFS`. Each box is represented by a chained list and is attached to a so-called set header cell. These cells are themselves kept in a chained list to establish the ordering of the boxes. Since we want to partition at most one box each time, we have to know which is the leftmost one. Each set header cell (each being the head of one box) is additionally furnished with two numbers, the total number of elements (vertices) in the boxes to its left and the number of elements in its attached box. The box to be partitioned has the smallest total number of vertices in boxes to its left. The partition operation has to create a new box and update two numbers. For every selected vertex, this takes time linear in its degree, which adds up to linear time in total. ∎

We want to characterize orderings that can be generated by `min-LexBFS`. A vertex ordering σ has *property (P1)* if and only if for every vertices $u, v \in V$ holds

$$\mathrm{lm}(v) \prec_\sigma u \prec_\sigma v \ \wedge\ uv \notin E \implies \bigvee_{w \in N(u)} \mathrm{lm}(v) \prec_\sigma \mathrm{lm}(w) \ \wedge\ v \prec_\sigma w.$$

It is evident that $\mathrm{lm}(w) \preceq_\sigma u$ must hold. We show that this property exactly characterizes min-LexBFS-orderings.

Proposition 11. *Let $G = (V, E)$ be a connected graph. A vertex ordering σ for G is a min-LexBFS-ordering for G if and only if σ has property (P1).*

Proof: Let σ be a min-LexBFS-ordering for G; let u and v, $u \prec_\sigma v$, be two vertices complying the premise. Then, $\mathrm{lm}(v)$ and u cannot be contained in the same box when v is chosen by the algorithm. Otherwise, the partition of the box containing $\mathrm{lm}(v)$ and u forced by v would put u into the left, $\mathrm{lm}(v)$ into the right partition box, which contradicts $\mathrm{lm}(v) \prec_\sigma u$. Hence, there is a neighbour w of u, $v \prec_\sigma w$, that separates $\mathrm{lm}(v)$ and u. Obviously, $\mathrm{lm}(w)$ is in the right partition box, like u. Therefore, $\mathrm{lm}(v) \prec_\sigma \mathrm{lm}(w)$.

Now, let σ be an ordering of the vertices of G fulfilling property (P1). Suppose, σ cannot be generated by `min-LexBFS`. Then, there is a first vertex x that separates two vertices u and v, $v \prec_\sigma u$, such that u is in the left partition box whereas v is in the right one. Without loss of generality, we may assume $v = \mathrm{lm}(x)$ by the choice of x. Obviously, u can never be chosen before v by `min-LexBFS`. By property (P1), there is a neighbour w of u, $x \prec_\sigma w$, that is not a neighbour of v. By the choice of x, u and v are contained in one box, when w is chosen. But w entails a partition such that v is in the left, u in the right partition box, which contradicts the assumption. ∎

Corollary 12. *An interval ordering is a min-LexBFS-ordering.*

Proof: Let $G = (V, E)$ be a graph and σ an interval ordering for G. Let u and v be two vertices, $\mathrm{lm}(v) \prec_\sigma u \prec_\sigma v$. By definition of σ, u and v are adjacent, so the premise of property (P1) is false. ∎

We already discussed the case of defining a start vertex for a `min-LexBFS` run. One step beyond, we tell the algorithm how to break ties in every case. Let `min-LexBFS*` be a variant of `min-LexBFS` that gets input G and σ where σ is a vertex ordering for G, and in each step `min-LexBFS*` chooses that vertex from the rightmost box that is leftmost in σ. We could say that the chosen vertex has highest preference with respect to σ. We define a *2min-LexBFS-ordering* for G to be the output of `min-LexBFS*(`G`, min-LexBFS(`G`))`, i.e., compute a min-LexBFS-ordering on the input graph and use this to break ties in a second run. It should be obvious how a $k$min-LexBFS-ordering is defined. If necessary, these algorithms are called k`min-LexBFS`.

Proposition 13. *A kmin-LexBFS-ordering can be computed in linear time.*

Proof: Let $G = (V, E)$ be a graph, and let σ be a vertex ordering for G. It is sufficient to show that `min-LexBFS*` runs in linear time. If we look closely at the implementation of `LexBFS` presented in [42], the vertices in the right partition box appear in the same order as they are kept in the adjacency list of the vertex that caused the partition (i.e., of the currently chosen vertex). It is no problem to always select the first vertex in the list of a box. In linear time, we can order the vertices in the adjacency lists of G according to σ. The vertices in the box V initializing the list (line 2) are ordered according to σ, too. ∎

4. Moplexes and the min-LexBFS algorithm. For a graph it may happen that two of its vertices have the same (open or closed) neighbourhood. Such two vertices are often called *twins*. This notion is strong enough to be extended to the basic notion of a *module*.

Definition 1. *Let $G = (V, E)$ be a graph, let $M \subseteq V$. The set M is a* ***module*** *in G if and only if for all vertices $u, v \in M$, $N_G(u) \setminus M = N_G(v) \setminus M$.*

So, every two vertices of a module have the same neighbourhood outside the module. Since every vertex itself and the whole vertex set are modules, every graph has modules. If a module M is of size 0, 1, or n, we call it *trivial*; M is a *maximal* module if it is not contained in another module. Modules were discovered in the literature several times independently, and Gallai showed that each graph can uniquely be partitioned into its maximal modules of size less than n up to isomorphism [17]; the iterative process on the maximal modules is called a modular decomposition. Further information about modules and modular decomposition can be found in [5] and [32].

In the late sixties of the last century, Roberts investigated proper interval graphs, which he also called indifference graphs [40]. He implicitly defined the notion of a *maximal clique-module* which was extended by Berry and Bordat to the notion of a *moplex* [3].

Definition 2. *Let $G = (V, E)$ be a graph, $M \subseteq V$. M is a* ***clique-module*** *if and only if M is a module as well as a clique; if M is not properly contained in another clique-module of G, M is a* ***maximal*** *clique-module. M is a* ***moplex*** *if and only if M is a maximal clique-module and its neighbourhood $N_G(M)$ is a minimal separator.*

For simplicity, if G is a clique, the whole vertex set V is a moplex (and the only one) of G. The relation *belonging to the same maximal clique-module* over the vertices of a graph is an equivalence relation [40]. Therefore, different moplexes are disjoint. The notion of moplexes seems to be a strong concept in graph theory and facilitates insight into the structure of graphs. A moplex is *simplicial* if its neighbourhood is a clique. Then, a graph is chordal if and only if one can repeatedly remove a simplicial moplex from the graph until the graph is empty [3]. Two moplexes are *adjacent* if one moplex contains a neighbour of the other; adjacent moplexes form a clique. We give a new characterization of chordal graphs in terms of moplexes.

Fact 14. *A graph $G = (V, E)$ is chordal if and only if there is no subgraph of G that contains two adjacent moplexes.*

Proof: In a chordless cylce of length at least 4 every maximal clique contains two moplexes. And since a graph is chordal if and only if every subgraph of it is chordal, it suffices to show that no chordal graph contains adjacent moplexes. Let $G = (V, E)$ be a chordal graph. Suppose M_1 and M_2 are adjacent moplexes and $C = M_1 \cup M_2$. Then, there is a vertex $u \in N(M_2)$ that is not contained in M_1 or a neighbour of M_1. Otherwise, $N(M_1)$ were not a minimal separator. $N(M_2)$ cannot be a clique, which contradicts G being chordal. ■

Further results involving moplexes are the existence of two non-adjacent moplexes in a graph that is not a clique, and the algorithm `LexBFS` always numbers a vertex last that belongs to a moplex of the input graph [3]. We call a vertex a *moplex vertex* if it belongs to a moplex.

In the following, we will show that also for min-LexBFS-orderings, the first vertex belongs to a moplex in the graph. To proof this we will follow the proof of the related result for LexBFS-orderings in [3]. A module $M \subseteq V$ in G is *connected* if $G[M]$ is connected.

Lemma 15. *Let $G = (V, E)$ be a connected graph, and let σ be a min-LexBFS-ordering for G. Let $a = \sigma(1)$ and $z = \sigma(n)$.*

i. Let $M = \{\sigma(1), \ldots, \sigma(i)\}$, $i \in \{1, \ldots, n-1\}$, be a connected module in G. Then, the ordering $\sigma' = \langle\sigma(1), \ldots, \sigma(i)\rangle$ is a min-LexBFS-ordering for $G' = G[M]$.

ii. Let $G-z$ be disconnected, and let C be a connected component of $G-z$. Then, σ' originating from σ by restriction to the vertices of $C+z$ is a min-LexBFS-ordering for $G' = C+z$.

iii. Let v be a vertex to the right of a such that for every vertex w, $v \prec_\sigma w$, $w \in N_G(a) \Leftrightarrow w \in N_G(v)$. Then, $M = \{a, \sigma(2), \ldots, v\}$ is a module in G.

iv. Let P be a u, z-path, $a \prec_\sigma u$, not containing a neighbour of a except for u. Then, there is a u, z-path $P' = (u, u_1, \ldots, u_\ell, z)$ not containing a neighbour of a except for u and $u \prec_\sigma u_1 \prec_\sigma \ldots \prec_\sigma u_\ell \prec_\sigma z$.

Proof: We will prove i and ii together. Let $u, v \in V(G')$, $uv \notin E$, and $\mathrm{lm}_{G'}(v) \prec_\sigma u \prec_\sigma v$. Obviously, $\mathrm{lm}_{G'}(v) = \mathrm{lm}_G(v)$, and there is a neighbour w of u in G to the right of v that is not a neighbour of $\mathrm{lm}(v)$. By definition, $w \in M$ since all neighbours of u to the right of $\sigma(i)$ are also neighbours of $\mathrm{lm}(v)$; this proves i. In case of ii, all neighbours of u in G are also contained in G', so w is contained in G' and to the right of v.

iii. Suppose there is a vertex u between a and v that has a neighbour to the right of v that is not a neighbour of v (and a). Then, v has a neighbour to its right that is not a neighbour of a which contradicts the premise of the statement. If v has a neighbour x to its right that is not a neighbour of a vertex u, $a \prec_\sigma u \prec_\sigma v$, u has a neighbour w to the right of x that is not a neighbour of a and therefore of v. Then, v has a neighbour to the right of w that is not a neighbour of a neither, which is a contradiction.

iv. Let x be the rightmost neighbour of a. All vertices u, $x \prec_\sigma u$, have such a desired u, z-path. Suppose there is $u \prec_\sigma x$ such that there is a u, z-path P not containing a neighbour of a except for u, but no such path is of the desired form; we choose u rightmost and assume P to be chordless. Then, the neighbour v of u in P is to the left of u, and there are two adjacent vertices b, c in P such that $b \prec_\sigma u \prec_\sigma c$. By property (P1) and $uc \notin E$, there is a neighbour w of u to the right of c that is not a neighbour of a since $\mathrm{lm}(c)$ is to the right of a, and by assumption there is a w, z-path of the desired form. Hence, there is a u, z-path P' of the desired form. ■

If a u, z-path in G is of the form of P', which means that starting at u we always go to a right neighbour with regard to σ, we will call it a *right-monotone* path. Of course, if we loosen the restiction of not passing neighbours of a these paths trivially exist by BFS-properties. From statement iii, we conclude that all vertices of a module containing a are numbered consecutively by `min-LexBFS`.

To prove that the first vertex of a min-LexBFS-ordering belongs to a moplex, we also have to find a minimal separator. Remember that a vertex set S in G is a minimal separator if and only if there are two components in $G \setminus S$ such that each vertex in S has neighbours in both components. By definition, every separator contains a minimal separator.

Lemma 16. *Let $G = (V, E)$ be a graph, and let $M \subseteq V$ be a non-trivial module in G. Let $S \subseteq M$ be a minimal separator in $G[M]$. Then, $S \cup N_G(M)$ is a minimal separator in G.*

Proof: Let S be a minimal separator in $G[M]$. Then, there are two components C and

D in $G[M] \setminus S$ that contain neighbours of every vertex in S. Furthermore, every vertex in $N(M)$ has a neighbour in C and D, so that every vertex in $S \cup N(M)$ has a neighbour in C and D. ■

Theorem 17. *Let $G = (V, E)$ be a connected graph, and let σ be a min-LexBFS-ordering for G. Then, $a = \sigma(1)$ belongs to a moplex M in G. Moreover, $M = \{\sigma(1), \ldots, \sigma(\ell)\}$, $\ell = |M|$.*

Proof: We will prove this by induction over the number of vertices of a graph. It is easy to see that the statement is true for all connected graphs on at most three vertices. Furthermore, it is true for all complete graphs. Let the statement be true for all graphs on at most $n-1$ vertices. Let $G = (V, E)$ be a connected graph on n vertices that is not a clique, and let $z = \sigma(n)$. If $G-z$ is disconnected and z is not universal, let C be the component of $G-z$ containing a, and let $G' = C+z$. Every subset of $N(a)$ is a minimal separator in G if and only if it is a minimal separator in G'. Then, by lemma 15 and the induction hypothesis, the statement is true for G' and hence for G. If z is universal, let C be the component of $G-z$ that contains a. A set S is a minimal separator in $G-z$ if and only if $S \cup \{z\}$ is a minimal separator in G. Then, the statement is true for C by lemma 15 and the induction hypothesis, and therefore it is true for G.

Now, let z be not universal, and let $G-z$ be connected. Let v be the rightmost vertex with respect to σ fulfilling $N[a] = N[v]$; to emphasize, a and v are adjacent, in fact, $a = \mathrm{lm}(v)$. Let $M = \{a, \sigma(2), \ldots, v\}$. Then, M is a module by lemma 15. If, for some u, $a \prec_\sigma u \prec_\sigma v$, u is not adjacent to v there must be a neighbour of u to the right of v that is not a neighbour of a, contradicting M being a module. If there are two non-adjacent vertices $u, w \in M$, $a \prec_\sigma u \prec_\sigma w \prec_\sigma v$, u must have a neighbour x to the right of w that is not a neighbour of a. Then, x cannot be to the right of v by M being a module, and x cannot be to the left of v since x is adjacent to a; a contradiction. Hence, M is a maximal clique-module.

Let b be the rightmost neighbour of a with respect to σ; $b \in N(M)$. Let x be the leftmost vertex for which there is an x, z-path P in G containing no neighbour of a except for x itself; $v \prec_\sigma x \preceq_\sigma b$. By lemma 15, we assume P to be right-monotone. Then, all vertices u between x and b have a u, z-path not containing a neighbour of a except for u. Suppose there is a vertex u, $x \prec_\sigma u \prec_\sigma b$, for which there is not such a u, z-path; let u be rightmost with this property, and let c, d be adjacent vertices on P, $c \prec_\sigma u \prec_\sigma d$. If $ud \in E$, we have an appropriate u, z-path; if $ud \notin E$, there is a neighbour w of u to the right of d that is not a neighbour of a. If w is to the right of b, we have found a suitable path, if w is to the left of b, there is a w, z-path not containing a neighbour of a. If there is no neighbour of a between v and x, $N(M)$ is a minimal separator in G.

Now, we show that all vertices to the left of x are contained in the same module. Suppose there is a vertex u to the left of x that has a neighbour to the right of x or x itself that is not a neighbour of a. Then, there is a u, z-path containing no neighbour of a except for u, contradicting the choice of x. Suppose there is a neighbour w of a, $u \prec_\sigma w$, that is not a neighbour of u. Then, u has a neighbour to the right of w that is not a neighbour of a; contradiction. Therefore, $M' = \{u : u \prec_\sigma x\}$ is a module. Then, $N(M)$ is a minimal separator in G if $N(M) \cap M'$ is a minimal separator in $G[M']$ by lemma 16. By lemma 15 and induction hypothesis, we conclude the theorem. ■

Although this result may seem to be only a consequence of the corresponding theorem for `LexBFS` at first glance, we draw attention to the graph depicted in figure 3. This graph

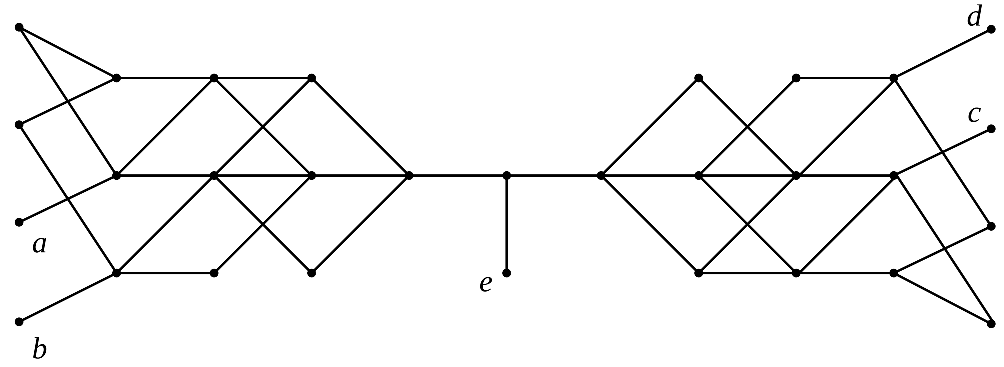

Figure 3. Do `LexBFS` and `min-LexBFS` generate the same moplexes?

contains exactly five moplexes labeled by the letters a, b, c, d, e. Independent of the start vertex, `LexBFS` can generate orderings with only b or d as first vertex, whereas `min-LexBFS` numbers only a or c last. Moplex $\{e\}$ can be generated by neither of the algorithms.

In [40], it is already mentioned that every chordal graph that is not a clique contains two non-adjacent moplexes. In case of non-complete interval graphs, two are always easy to find: given graph G, run `LexBFS` or `min-LexBFS` on G, which generates one moplex by theorem 17, and rerun the algorithm on G starting with the last numbered vertex of the first run. Since G is not complete, the last numbered vertex in the second run cannot be adjacent to the start vertex and must therefore be contained in a second moplex. If G is a proper interval graph, G contains at most two moplexes [40]. But this is not a characterization of proper interval graphs. Take as an example the P_5 plus a universal vertex, which is an interval graph containing exactly two moplexes (the endvertices of the P_5), but is not a proper interval graph since it contains a claw.

Corollary 18. *The first vertex of an interval ordering is a moplex vertex.*

Proof: This is an immediate consequence of corollary 12 and theorem 17. ∎

5. Certifying recognition of proper interval graphs. In the two preceeding sections we introduced the new algorithm `min-LexBFS`, characterized orderings that may be generated by `min-LexBFS`, and proved that each min-LexBFS-ordering starts with a moplex vertex. In this section, we aim at providing the first application of `min-LexBFS`. We will present an easy linear-time algorithm that recognizes proper interval graphs. A 3-sweep `min-LexBFS` (`3min-LexBFS`) suffices to generate a proper interval ordering for a given graph if and only if it is a proper interval graph. Other recognition algorithms are known that can be found in [7], [24], or [13]. They all base more or less directly on the computation of a proper interval ordering.

In the area of graph recognition algorithms, recent developments focus on an extension of the task of simply recognizing a class of graphs. A classical decision algorithm normally provides answer "yes" or "no", and the user can do nothing more than believing that the answer is correct. To improve the user's life, the concept of a *certifying* algorithm was introduced. Such an algorithm provides a certificate in both possible cases, i.e., if the input graph belongs to the graph class ("has the desired property") or if it does not. There is no formal definition of a certificate, but intuitively spoken, it should be easily verifiable. Only a small number of graph classes provide an efficient and certifying recognition algorithm among which is the well-known recognition algorithm for bipartite graphs. Start a breath first search and check whether every level is an independent set. If so, the algorithm may easily produce a 2-colouring of the graph; if not, it easily finds a cylce of odd length. Both certificates can be verified in linear time. Other examples of certifying recognition algorithms are presented in [28] for interval and permutation graphs.

In our case, proper interval graphs provide almost canonical certificates. If a given input graph G is a proper interval graph, there is a proper interval ordering of its vertices. On the other hand, if G is not a proper interval graph, it must contain a chordless cycle of length at least 4, an asteroidal triple, or a claw as induced subgraph. Applying, for instance, lemma 5, we can verify in linear time, if a given vertex ordering is a proper interval ordering for G. If four vertices induce a claw in G, if three vertices form an asteroidal triple, or if a sequence of vertices induces a chordless cycle can also be verified in linear time. These certificates are "good" certificates. Up to our knowledge, the recognition algorithm presented here is the only one for proper interval graphs that is certifying.

Before dealing with the algorithm itself, we will prove some interesting properties of min-LexBFS-orderings of AT-free and interval graphs. It will be essential to the correctness of our algorithm to know that the distance between the first and the last vertex of a 2min-LexBFS-ordering of an interval graph is equal to the diameter of this graph. First, we show what the linear structure of AT-free graphs (cf. [8] or [10]) means to 2BFS-orderings. A double sweep `BFS` is defined as follows. Start a first BFS run at any vertex, and for the second sweep take as start vertex a vertex from the last level of the first run. Orderings that can be generated by `2BFS` are thus called 2BFS-orderings.

Lemma 19. *Let $G = (V, E)$ be a connected graph. Let σ be a 2BFS-ordering for G, and let $S_0, \dots, S_k$, $k \geq 3$, be the BFS-levels of G with root $z = \sigma(n)$. Then, $G' = G \setminus S_1$ contains at most one component with vertices from S_3. If G is AT-free, then for every $i \geq 2$, $G \setminus S_{i-1}$ contains at most one component with vertices from S_{i+1}.*

Proof: We start with the first statement of the lemma. Suppose there are two components in G' that contain vertices from S_3. Since z is in distance 1 to every vertex in S_1, the first sweep of `2BFS` could not have numbered first a neighbour of z (S_3 is not empty). Therefore, the start vertex s of the first sweep must be contained in S_i for some $i \geq 2$; $d(z, s) = i$. By our assumption, there is a vertex $x \in S_3$ that is not in the same component of G' as s, so that $d(s, x) \geq i - 1 + 2 = i + 1 > i = d(z, s)$. But then, z could not be the vertex numbered last during the first BFS sweep. The assumption is false. Now, for the second statement, suppose there is some $i \geq 3$ such that $G \setminus S_{i-1}$ contains two component C_1 and C_2 with vertices from S_{i+1}. Let $u, v \in S_{i+1}$, and u is in C_1 whereas v is in C_2. The neighbourhoods of u and v are completely contained in C_1 and C_2, respectively. But then, u, v, and z form an asteroidal triple in G, which contradicts G being AT-free. ■

The first part of the proof shows that, if $k \geq 3$, the first BFS sweep must have started on a vertex of C, the only component of G' containing vertices from S_3. Otherwise, z could not be the last numbered vertex during the first sweep.

Lemma 20. *Let $G = (V, E)$ be a connected graph that is not a clique. Let σ be a 2min-LexBFS-ordering for G, and let $S_0, \dots, S_k$ be the BFS-levels of G with root $z = \sigma(n)$. G has a universal vertex, or there is a component C of $G \setminus S_1$ that contains a vertex x whose neighbourhood in S_1 is a subset of the neighbourhood of every vertex in S_2 that is not contained in C. Furthermore, if G is AT-free, then for every i, $2 < i < k$, if D_i is the component of $G \setminus S_{i-1}$ that contains a vertex from S_{i+1} and $u \in S_i$ is not contained in D_i, it holds that $N_G(D_i) \subseteq N_G(u)$.*

Proof: Let σ' be the ordering generated by the first sweep of `2min-LexBFS`, and let $s = \sigma'(n)$. If s and z are adjacent, then s is a universal vertex in G. Otherwise, s is not contained in S_1. Let C be the component of $G \setminus S_1$ containing s. Lemma 19 ensures that C contains all vertices that are not contained in $S_0 \cup S_1 \cup S_2$. Let x be the righmost vertex from S_2 with respect to σ'. If S'_ℓ is the last BFS-level corresponding to σ', x is contained in $S'_{\ell-2}$. All vertices from S_2 that are not contained in C appear in S'_ℓ. Suppose there is a vertex u in a component C' of $G \setminus S_1$ that is not adjacent to some neighbour $v \in S_1$ of x. By construction, it holds that $z \prec_{\sigma'} u \prec_{\sigma'} v \prec_{\sigma'} x$. There must be a neighbour of u to the right of v that is not a neighbour of z by property (P1). But since every vertex from S_1 to the right of v is a neighbour of z this is not possible. Therefore, $N(x) \cap S_1 \subseteq N(u)$ for every vertex $u \in S_1$ that is not contained in C. Now, let G be AT-free. Suppose there is some i such that D_i contains a vertex from S_i that has a neighbour a in S_{i-1}, and a and u are not adjacent for some $u \in S_i$ not contained in D_i. Let v be any vertex in S_{i+1}. First, u and v are in the same component of $G \setminus N(z) = G \setminus S_1$ by lemma 19. Since a and v are in one component of $G \setminus N(u)$, the vertices u, v, and z form an asteroidal triple in G which contradicts the assumption. ■

Lemma 21. *Let $G = (V, E)$ be a connected AT-free graph. Let σ be a min-LexBFS-ordering for G, and let $S_0, \ldots, S_k$, $k \geq 3$, be the BFS-levels of G with root $z = \sigma(n)$. Let C be a component of $G \setminus S_1$ containing vertices from S_3. Let $a_i \in S_i$, $i \geq 3$, be the leftmost vertex in C with respect to σ. Then, every neighbour of a_i in S_{i-1} is adjacent to every non-neighbour of a_i in S_i contained in C.*

Proof: Suppose there is an $a_i \in \{3, \ldots, k\}$ such that there is a vertex $u \in S_i$ that is in C and not adjacent to a_i and a neighbour $x \in S_{i-1}$ of a_i. By property (P1), there must be a neighbour $w \in S_{i-1}$ of u that is not adjacent to a_i. By assumption, u and a_i are contained in one component of $G \setminus S_1$, so that a_i, u, and z would form an asteroidal triple in G. ■

From the last two lemmata above, we will conclude that each level of a 2min-LexBFS-ordering of an interval graph contains a vertex that is adjacent to every vertex in the level below. Let σ be a 2min-LexBFS-ordering of a connected interval graph $G = (V, E)$, and let $S_0, \ldots, S_k$ be the corresponding BFS-levels. Let $u, v \in S_i$, $i \geq 2$, be contained in one component C of $G \setminus S_{i-1}$. Note that $N(C) \subseteq S_{i-1}$ is a minimal separator in G and therefore a clique. Suppose u has a neighbour in S_{i-1} that is not a neighbour of v and v has a neighbour in S_{i-1} that is not a neighbour of u. If u and v are adjacent, they are contained in a chordless cycle of length 4. If u and v are not adjacent, u, v and $\sigma(n)$ form an asteroidal triple. Therefore, $N(u) \cap S_{i-1} \subseteq N(v)$ or $N(v) \cap S_{i-1} \subseteq N(u)$. Then, all vertices of C in S_i have a common neighbour. This immediately entails the property of 2min-LexBFS-orderings for interval graphs that the first and the last vertex in the ordering are vertices of maximal distance in the graph.

Proposition 22. *Let $G = (V, E)$ be a connected interval graph. Let σ be a 2min-LexBFS-ordering for G, and let $a = \sigma(1)$ and $z = \sigma(n)$. Then,* $\mathrm{diam}(G) = d(z, a)$.

Proof: Let $S_0, \ldots, S_k$ be the BFS-levels of G with root z. If $k \leq 1$, then G is a clique, and the statement holds. Let $u, v \in S_i$, $i \in \{2, \ldots, k\}$, be contained in one component of $G \setminus S_{i-1}$. Then, $N(u) \cap S_{i-1} \subseteq N(v)$ or $N(v) \cap S_{i-1} \subseteq N(u)$. If $k = 2$, G contains a universal vertex in S_1 by lemma 20. Let $k \geq 3$, and let $u \in S_i$, $v \in S_j$, $1 \leq i \leq j \leq k$, $i < k$. By BFS-properties, there is a vertex $x \in S_i$ such that $d(x, v) = j - i$. If $i > 1$, there is a common neighbour of u and x in S_{i-1} by lemma 20 and the discussion above. We obtain $d(u, v) \leq d(x, v) + 2 = j - i + 2 \leq k$. If $i = 1$, u and x are adjacent, since z is a moplex vertex, which gives $d(u,v) \le d(x,v) + 1 = j - i + 1 \le k$. Let $i = j = k$. By lemma 21, u and v have distance at most 2. ■

Now, we are well-prepared to look at the recognition of proper interval graphs. We will first prove a lemma which, in fact, is the central part of the correctness of our algorithm.

Lemma 23. *Let $G = (V, E)$ be a connected graph, and let σ be a 3min-LexBFS-ordering for G. If σ is not a proper interval ordering, then G contains an AT, a claw, or a chordless cycle of length at least 4, one of which can be found in linear time.*

Proof: Let $z = \sigma(n)$, and let $S_0, \ldots, S_k$ be the BFS-levels defined by σ. If σ is not a proper interval ordering, there must be vertices $u, v, w \in V$, $u \prec_\sigma v \prec_\sigma w$, $uw \in E$, such that $uv \notin E$ or $vw \notin E$. If σ is not an interval ordering there must be a rightmost vertex w such that there is a rightmost vertex v, $u \prec_\sigma v \prec_\sigma w$, $u = \mathrm{lm}(w)$, with $vw \notin E$. Let $v \in S_j$ and $w \in S_i$. Since $uw \in E$, $1 \le i \le j \le i+1$. Let $i = j = 1$. Since z is a moplex vertex, v and w have neighbours in S_2. If they have a common neighbour a, $\{a, v, w, z\}$ induces a chordless cycle of length 4 in G. If they do not have a common neighbour in S_2, v and w have neighbours in one connected component of $G' = G \setminus N_G[z]$, and there is a shortest v, w-path P in $G' + v, w$. We obtain a chordless cycle $P + z$ of length at least 5. Let $i = j \ge 2$. Let $x \in S_{i-1}$ be a neighbour of v. By the choice of v and w, x is a neighbour of w and has itself a neighbour $a \in S_{i-2}$, so that $\{v, w, x, a\}$ induces a claw in G. Let $1 \le i < j$. By property (P1), there is a neighbour $a \in S_i$ of v, $w \prec_\sigma a$, that is not a neighbour of u. By the choice of u, v, and w, w and a are adjacent. If $uv \in E$, $\{u, v, w, a\}$ induces a chordless cycle of length 4. Let $uv \notin E$. If $j \ge 3$, then u, v, z form an AT. Let $j = 2$. If u and v are contained in the same component of $G \setminus S_1$, then u, v, z form an AT. If they are not contained in one component, a or w (being neighbours of z) must have neighbours in two different components of $G \setminus S_1$ (by the moplex property). Then, we find a claw in G.

Now, if σ is an interval ordering, there is a leftmost vertex $u \in V$, its rightmost neighbour w, and a leftmost vertex v, $u \prec_\sigma v \prec_\sigma w$, such that $uv \notin E$. If there is a neighbour x of w to the right of w that is not a neighbour of v, then $\{u, v, w, x\}$ induces a claw in G. In the remaining part of the proof, we will show that $N^+(w) \subseteq N(v)$ is not possible, which completes the proof. Let $a = \sigma(1)$. Let $w \in S_i$, then, $u \in S_{i+1}$. By the assumption $N^+(w) \subseteq N(v)$, v is also in S_i. We first observe that $\mathrm{lm}(w) = u$ since a neighbour of w to the left of u must also be a neighbour of v which implies v being a neighbour of u. If there is a shortest a, z-path using a vertex in S_{i+1} to the right of u, then $d(a, u) = k - i$; if there are only shortest a, z-paths using a vertex in S_{i+1} to the left of u, then $d(a, u) = k - i + 1$. It follows by the choice of u and v that $d(a, w) = d(a, u) + 1 \le d(a, v)$. Let S be the set of all vertices in distance $d(a, u)$ to a; either $S \subseteq S_i \cup S_{i+1}$ or $S \subseteq S_{i+1} \cup S_{i+2}$. Let S' be the set of vertices in distance $d(a, w)$. Every vertex in S that is a neighbour of w is a vertex to the right of u with respect to σ and must therefore be a neighbour of every other neighbour of u in S'. It is also clear that a neighbour of v in S is also a neighbour of w. Since a has a neighbour in S_{k-1} and v is not in S_k, S_k induces a clique. By fact 14, a is contained in the only moplex M in G in maximal distance to z. By theorem 17 and proposition 22, the last numbered vertex of the first sweep is contained in M, and since vertices in a moplex can be numbered by `min-LexBFS` in any order, a must have had the least priority during the third sweep. In fact, the second sweep must have started at a. Consider the second sweep of `3min-LexBFS`. Before a vertex in S is chosen by the algorithm, all vertices in S' remain in the leftmost box. Until a non-neighbour of v is selected, v and

w remain in the leftmost box containing a neighbour of u. After the selection of u, v and w have finally been separated. So, for every min-LexBFS-ordering with last vertex a, v appears to the left of w. During the final, third, sweep of `3min-LexBFS`, v must be selected before w, which contradicts the assumption of $N^+(w)$ being a subset of $N(v)$.

It is an easy exercise to implement the described algorithm to run in linear time and find an asteroidal triple, a chordless cycle, or a claw. ∎

Theorem 24. `3min-LexBFS` *is a linear-time certifying recognition algorithm for proper interval graphs.*

Proof: It is clear that a vertex ordering of a disconnected graph is a 3min-LexBFs-ordering if and only if the ordering is a 3min-LexBFS-ordering for each component (which means for each component, the generated ordering is restricted only to the vertices of this component). Furthermore, no vertex of one component appears between two vertices of another component. Now, let G be a graph and σ a 3min-LexBFS-ordering of G. In linear time, we can check whether σ is a proper interval ordering for G, for instance by applying lemma 5. If so, G is a proper interval graph, and σ is a certificate. If not, we apply (the algorithm described in) lemma 23 to every component of G and the corresponding part of σ and obtain in linear time a certificate for G not being a proper interval graph. ∎

A similar approach based on a multi-sweep `LexBFS` has been used by Corneil, Olariu, and Stewart to recognize interval graphs [9]. They show that mainly four sweeps of `LexBFS` suffice to generate an interval ordering for a given input graph G if and only if G is an interval graph. In the case of proper interval graphs, Corneil independently showed that a 3-sweep `LexBFS` outputs a proper interval ordering if and only if the input graph is a proper interval graph [6].

6. On minimal triangulations of AT-free claw-free graphs. A graph is AT-free claw-free if it does not contain an AT and a claw as induced subgraphs. The cycle on five vertices is AT-free and claw-free as well as every co-bipartite graph. These graphs have been studied before in the literature by several authors [39], [27], [23]. They allow faster algorithms for a number of famous graph problems compared to the whole class of AT-free graphs [23]. Part of this is based on the following structural result which was already given by Hempel and Kratsch [23] in a weaker version.

Lemma 25. *Let $G = (V, E)$ be a connected AT-free claw-free graph; let z be a moplex vertex in G. Let $S_0, \ldots, S_k$ be the BFS-levels of G with root z. Then, the following holds:*

i. $S_0, S_2, \ldots, S_k$ induce cliques in G;

ii. for every two non-adjacent vertices $u, v \in S_1$, u and v have neighbours in S_2.

Proof: Statement ii. is an immediate consequence of z being a moplex vertex since u and v do not belong to the moplex containing z.

i. Suppose there are two non-adjacent vertices $u, v \in S_i$, $i \in \{2, \ldots, k\}$; if u and v have a common neighbour w in S_{i-1} which must have a neighbour x in S_{i-2}, the vertices u, v, w, x induce a claw in G. Now, u and v do not have a common neighbour. Let i be smallest possible; let M be the moplex containing z. If $i = 2$ and u and v are contained in the same component of $G \setminus N(M)$, there is a u, v-path in G avoiding the neighbourhood of z. Since u and v do not have a common neighbour in S_1, u, v, z form an AT contradicting G being AT-free. If $G \setminus N(M)$ contains three components, every vertex in $N(M)$ can have neighbours in at most two of them, since otherwise there would be a claw in G. But then,

$N(M)$ cannot be a minimal separator in G. If $i > 2$, u, v, and z form an AT in G, since u and v do not have a common neighbour in S_{i-1}, and there is a u,v-path of length 3 using vertices from S_{i-1}, thus avoiding the neighbourhood of z. ■

In [23], z was to be the last numbered vertex of a LexBFS-run on G. Since this is always a moplex vertex [3], the result in [23] is an immeadiate corollary of lemma 25.

Corollary 26. **[23]** *Let $G = (V, E)$ be a connected AT-free claw-free graph; let $z \in V$ be a moplex vertex of G. Let $S_0, \ldots, S_k$ be the BFS-levels of G with root z. Then,* $\mathrm{diam}(G) = k$.

Proof: Using lemma 25, the proof is equivalent to the proof in [23]. ■

In section 3, `2min-LexBFS` was defined, which is a double run of `min-LexBFS` on the input graph but picking vertices during the second run with regard to the min-LexBFS-ordering generated by the first run. We first show that `2min-LexBFS` always outputs a minimal interval elimination scheme for a connected AT-free claw-free graph as input. A minimal triangulation of such a graph is always a proper interval graphs [39], [27]. Remember that, by lemma 5, an interval model of the filled-in graph can be computed in linear time. If an ordering σ is a BFS-ordering with last vertex z, we associate with σ the BFS-levels $S_0, \ldots, S_k$.

Lemma 27. *Let $G = (V, E)$ be a connected AT-free claw-free graph; let σ be a 2min-LexBFS-ordering for G. Then, σ is a minimal interval elimination scheme for G.*

Proof: Let $H = (G, \sigma)$, and let $z = \sigma(n)$. First, we show that σ is an interval ordering for H. By definition of H, the rightmost vertex in S_2 is a neighbour of all vertices in S_1 that have a neighbour in S_2. Therefore, S_1 is a clique in H, hence every S_i is a clique in H. Suppose there are vertices $u, v, w \in V$, $u \prec_\sigma v \prec_\sigma w$, such that $uw \in E(H)$ and $vw \notin E(H)$. Then, v and w are not in the same level, and $uv \in E$. By σ being a min-LexBFS-ordering of G, there is a neighbour x of v in the same level as w (and therefore adjacent to w) that is not a neighbour of u. Then, (u, v, x, w) is a chordless cycle of length 4 in H, which contradicts H being chordal. To show that σ is a minimal elimintation scheme for G, we have to show for every edge in $F = E(H) \setminus E$ that it is unique chord in a cycle of length 4 in H (theorem 2). Let $uv \in F$, $u \prec_\sigma v$, $u \in S_j$, $v \in S_i$, $i \leq j$. If $i = j$, then $i = j = 1$ by lemma 25. If u and v have a common neighbour $x \in S_2$, x and z are not adjacent in H by lemma 5, and uv is unique chord in the cycle (u, x, v, z). Otherwise, let $w \in S_2$ be a neighbour of u, $x \in S_2$ be a neighbour of v. Since H is chordal, $ux \in F$ or $vw \in F$ must hold. Then, uv is unique chord in the cycle (u, x, v, z) or (u, w, v, z). Let $j = i{+}1 \geq 2$. Then, $\mathrm{lm}(v)$, $\mathrm{lm}(v) \prec_\sigma u$, is a common neighbour of u and v in H by lemmata 5 and 25. Applying proposition 11, u has a neighbour w to the right of v that is not adjacent to $\mathrm{lm}(v)$ in H, and by $w \in S_i$, $vw \in E$. Then, uv is unique chord in the cycle $(\mathrm{lm}(v), u, w, v)$. Finally, $j > i{+}1$ cannot be true since the leftmost neighbour of v in H is in S_{i+1} or S_i by lemma 5. ■

We have to make one remark on the proof. If $j = i{+}1 = 2$ in the second case, it is true that v and w do not need to be adjacent in G due to lemma 25. But the first case $(i = j)$ shows that they must be adjacent in H.

Corollary 28. *Let $G = (V, E)$ be a connected proper interval graph, and let σ be a 2min-LexBFS-ordering for G. Then, $G = (G, \sigma)$.*

Proof: Let G be a proper interval graph. By lemma 27, (G, σ) is a minimal triangulation of G, and since G is chordal, $G = (G, \sigma)$. ■

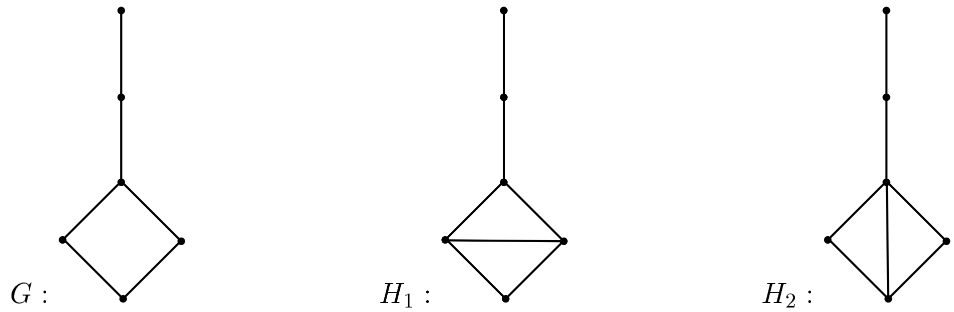

Figure 4. Graph G and its two different minimal triangulations H_1 and H_2.

Though, corollary 28 does not imply a recognition algorithm for proper interval graphs since there are interval graphs that are not proper interval graphs for which the statement of the corollary also holds. The smallest such example is the claw.

In [3], it was shown that for a graph G and a minimal triangulation H of G, every moplex of H is a moplex of G with the same neighbourhood in both graphs.

Lemma 29. *Let $G = (V, E)$ be a connected AT-free claw-free graph, and let $H = G \cup F$ be a minimal triangulation of G. Let z be a moplex vertex of H, and let $S_0, \ldots, S_k$ be the BFS-levels with root z. Then, $S_0, \ldots, S_k$ are also the BFS-levels of G.*

Proof: Since $S_0 \cup S_1 = N_H[z] = N_G[z]$, the statement is true for S_0 and S_1. Let S_i, $i \geq 2$, be the first level for which the statement is not true; let S'_i be the ith level of G with root z. By H being a triangulation of G, $S'_i \subset S_i$. Let $u \in S_i \setminus S'_i$, which means that u does not have a neighbour in S_{i-1} in G. We can delete all edges between u and vertices in S_{i-1} in H and still obtain a chordal graph H'. Otherwise, there must be a cycle C of length at least 4 containing u. Easy to see that C can contain only two vertices from S_i since S_i is a clique, and u should have a neighbour in S_{i-1}. Hence, H' is chordal contradicting H being a minimal triangulation of G. ∎

Corollary 30. *Let G be a connected AT-free claw-free graph, and let H be a minimal triangulation of G. Then,* $\mathrm{diam}(H) = \mathrm{diam}(G)$.

Proof: Since G is a subgraph of H, $\mathrm{diam}(G) \geq \mathrm{diam}(H)$. Let z be a moplex vertex of H, and let $S_0, \ldots, S_k$ be the BFS-levels of H with root z. By lemma 29, $S_0, \ldots, S_k$ are the BFS-levels of G with root z, and since z is a moplex vertex also in G, we obtain by lemma 26 that $\mathrm{diam}(G) = k \leq \mathrm{diam}(H)$. ∎

Figure 4 shows a co-comparability graph G and its two different minimal triangulations H_1 and H_2. Although H_1 has diameter 4, H_2 has diameter 3; G contains a claw. By the way, H_2 cannot be computed by `2min-LexBFS`, which means that no 2min-LexBFS-ordering σ of G yields $H_2 = (G, \sigma)$. But for σ being a 2min-LexBFS-ordering of H_2, $H_2 = (H_2, \sigma)$, which shows that corollary 30 is best possible, i.e., there are connected graphs that do contain a claw and all minimal triangulations have the same diameter. This is especially true for chordal graphs.

We already know how to compute a minimal triangulation of a connected AT-free claw-free graph G. It is done by a simple run of `2min-LexBFS` and the computation of the filled-in graph. We want to show that **every** minimal triangulation H of G can be computed by `2min-LexBFS`. Let σ be a proper interval ordering for H. Then, for every two adjacent vertices u, v, $u \prec_\sigma v$, it holds that

$$N_H^+(u) \subseteq N_H[v].$$

This is an immediate consequence of σ being a proper interval ordering: Let x be the rightmost neighbour of u, $v \neq x$; then $u \prec_\sigma v \prec_\sigma x$, and v is neighbour of u and x. Furthermore, all vertices between u and v and between v and x are neighbours of v, which shows the statement. Remember that every proper interval ordering is an interval ordering and therefore a BFS-ordering.

Lemma 31. *Let $G = (V, E)$ be a connected AT-free claw-free graph, and let $H = G \cup F$ be a minimal triangulation of G. Let σ be a proper interval ordering for H. Then, σ can be generated by a* `min-LexBFS` *run on G. Furthermore, $H = (G, \sigma)$.*

Proof: Let $S_0, \ldots, S_k$ be the BFS-levels of G with root $\sigma(n)$. We try to generate σ by a `min-LexBFS` run on G, i.e., we always chose the vertex determined by σ. Suppose that this is not possible. Then, there is a vertex x that separates vertices u and v, $u \prec_\sigma v$, $ux \in E$ and $vx \notin E$; let x be rightmost in σ. Without loss of generality, we may assume $u = \mathrm{lm}(x)$. Since `min-LexBFS` always computes a BFS-ordering, u and v must be contained in the same level S_i, $i \geq 2$, and therefore be adjacent ($i = 1$ is not possible since every vertex in S_1 not being adjacent to every other vertex in S_1 has a neighbour in S_2; x separates vertices not contained in $N_G(z)$). By properties of proper interval orderings, $N_H^+(u) \subseteq N_H[v]$, and hence $vx \in F$. Since H is a minimal triangulation of G, there are vertices a and b, $a \prec_\sigma b$, such that vx is unique chord in the cycle (a, v, b, x). Then, $a \prec_\sigma v \prec_\sigma x \prec_\sigma b$, and b cannot be a neighbour of u. Otherwise, a were a neighbour of b by the definition of a proper interval ordering. We conclude that there is a vertex to the right of x that separates u and v, which contradicts the assumption.

Suppose $(G, \sigma) \neq H$. Since H is a minimal triangulation of G, there must be an edge in (G, σ) that is not in H. Let $u \in S_i$ be the leftmost vertex with regard to σ that makes two of its neighbours, v and w, adjacent in (G, σ) that are not adjacent in H. Then, $i \geq 2$, since $S_0 \cup S_1$ induces a clique in H, $v \in S_i$, $w \in S_{i-1}$, $u \prec_\sigma v \prec_\sigma w$. Since σ is a proper interval ordering, $N_H^+(u) \subseteq N_H[v]$, and $vw \in F$. Contradiction. ■

The lemma also shows another interesting property of `min-LexBFS` on AT-free claw-free graphs. Let $G = (V, E)$ be an AT-free claw-free graph that is not a clique, and let $M \subseteq V$ be a moplex of G. By making $V \setminus M$ a clique we obtain a triangulation H' of G. There must be a minimal triangulation H of G such that $G \subseteq H \subseteq H'$, which contains M as a moplex. We know that each proper interval graph contains at most two moplexes. Furthermore, the first and the last vertex of a proper interval ordering for H belong to different moplexes by theorem 17 and corollary 18. Let σ be a proper interval ordering for H such that the last vertex of σ does not belong to M. Then, the vertices of M must be the first vertices of σ. Since σ is also a min-LexBFS-ordering for G, each moplex of an AT-free claw-free graph can be generated by `min-LexBFS`.

Theorem 32. *Let $G = (V, E)$ be a connected AT-free claw-free graph. A graph H is a minimal triangulation of G if and only if there is a 2min-LexBFS-ordering σ for G such that $H = (G, \sigma)$.*

Proof: If σ is a 2min-LexBFS-ordering of G, $H = (G, \sigma)$ is a minimal triangulation of G by lemma 27. If H is a minimal triangulation of G, it is a proper interval graph [39], and there is a proper interval ordering σ for H, which can be generated by `min-LexBFS` (lemma 31). The reverse ordering $\vec{\sigma}$ to σ is also a proper interval ordering for H and can therefore be generated by `min-LexBFS`, too, such that σ is a 2min-LexBFS-ordering for G, and $H = (G, \sigma)$ by lemma 31. ■

At the end of this section, we want to solve a decision problem related to computing a minimal triangulation of a graph. We show that we can decide in linear time whether a graph is a minimal triangulation of a connected AT-free claw-free graph. This question for an arbitrary graph G, whether a given chordal graph H is a minimal triangulation of it, is solvable in polynomial time, of course, even in time $\mathcal{O}(n^2 \cdot m)$, since for every edge $e \in E(H) \setminus E(G)$, we only have to check if $H - e$ is chordal. Note that it is necessary that the vertices of both graphs are equally named since otherwise the (general) problem is at least as hard as the graph isomorphism problem on chordal graphs.

Theorem 33. *Let $G = (V, E)$ be a connected AT-free claw-free graph; let H be a graph on the same vertex set as G. It is linear-time decidable whether H is a minimal triangulation of G.*

Proof: We describe a decision algorithm.

0. If H is not connected, return "No".
1. Let σ be a 3min-LexBFS-ordering of H.
2. If σ is not a proper interval ordering of H, return "No".
3. Run `min-LexBFS`* on input G and $\vec{\sigma}$; let σ' be the output.
4. If $\sigma \neq \sigma'$, return "No".
5. If $H = (G, \sigma)$ then return "Yes" else return "No".

Since every single step is executable in linear time, the algorithm runs obviously in linear time (by lemma 5, corollary 6, and proposition 13).

Correctness. To be a minimal triangulation of G, H has to be a connected proper interval graph [39], which is verified by steps 0, 1, 2 (cf. theorem 24). If H is a minimal triangulation of G, the algorithm outputs "Yes" by lemma 31. If H is not a minimal triangulation of G and step 4 nevertheless ends successfully, σ is a minimal interval elimination scheme for G by lemma 27, and therefore, (G, σ) is not equal to H. ■

In the literature, a more general problem has been studied. Given a graph G and an arbitrary triangulation H of G. How much does it cost to make H a minimal triangulation of G? One has to find a maximal set of edges of H that can be deleted and the resulting graph is still a triangulation of G. Blair, Heggernes, and Telle presented an algorithm that solves the problem in time $\mathcal{O}(f \cdot (m + f))$ where f denotes the number of edges in H that are not in G [4]. Another algorithm was given by Dahlhaus [12]; it runs in time $\mathcal{O}(nm)$ and improves the first result if f is of order at least n. If no edge has to be deleted H is already a minimal triangulation of G.

7. On minimal triangulations of co-comparability graphs and an application to the bandwidth problem. Co-comparability graphs are AT-free [20], and therefore every minimal triangulation of a co-comparability graph is an interval graph (theorem 3). In this section, we will show that `min-LexBFS` is also useful to compute a minimal triangulation of a co-comparability graph easily and efficiently. But in contrast to AT-free claw-free graph, the algorithm cannot compute all minimal triangulations of a given graph, more precisely, there are a co-comparability graph and a minimal triangulation of it that cannot be computed by any `min-LexBFS`-run, as we will see later.

Lemma 34. *Let $G = (V, E)$ be a connected co-comparability graph and σ a co-comparability ordering for G. Let $\tau =$ `min-LexBFS`$^*(G, \sigma)$. Then, for all non-adjacent vertices $u, v \in V$, $u \prec_\sigma v$ implies $v \prec_\tau u$.*

Proof: Suppose there are a rightmost vertex $v \in V$ with regard to τ and a vertex $u \in V$, $uv \notin E$, such that $u \prec_\sigma v$ and $u \prec_\tau v$. When v was selected by `min-LexBFS`*, u could not be contained in the same box as v. Then, there must be a vertex x, $xu \notin E$, $xv \in E$, that separated u and v. Since $xu \notin E$ and by the choice of v, x must be to the left of u in σ, which gives $x \prec_\sigma u \prec_\sigma v$. By the definition of a co-comparability ordering and $xv \in E$, u must be adjacent to x or v which contradicts the assumptions above. ∎

The following corollary states that the result of this process is again a co-comparability ordering of the input graph so that there is, for every (connected) co-comparability graph, a co-comparability ordering that is a fixed-point of `min-LexBFS`* except for the orientation of the ordering. This means that the generated vertex ordering is exactly the reverse of the fixed-point input ordering.

Corollary 35. *Let $G = (V, E)$ be a connected co-comparability graph and σ a co-comparability ordering for G. Then, $\tau =$ `min-LexBFS`*$(G,\ \sigma)$ is a co-comparability ordering for G.*

Proof: Suppose there are three vertices $u, v, w \in V$, $u \prec_\tau v \prec_\tau w$, such that $uw \in E$, and v is not adjacent to u and w. By lemma 34, $w \prec_\sigma v$ and $v \prec_\sigma u$, which contradicts σ being a co-comparability ordering for G. ∎

We immediately conclude that for a given connected co-comparability graph there is a co-comparability ordering that can be computed by `min-LexBFS`. A similar result was shown by Corneil, Olariu, Stewart in case of `LexBFS` [11]. An interesting question is if it is possible to guide `min-LexBFS` to always compute a co-comparability ordering running on a co-comparability graph. The same question has been asked in [11] for `LexBFS`.

Proposition 36. *Let $G = (V, E)$ be a connected co-comparability graph and σ a co-comparability ordering for G. Then, $\tau =$ `min-LexBFS`*(G, σ) is a minimal interval elimination scheme for G.*

Proof: Let $H = (G, \tau)$, and let $uv \in E(H) \setminus E$, $u \prec_\tau v$. By lemma 34, $v \prec_\sigma u$. Applying theorem 2, it suffices to show that uv is unique chord in a cycle of length 4 in H. By the definitions of H and `min-LexBFS`, when v was chosen there must be a neighbour z of v contained in a box to the left of the box containing u. Then, $uz \in E$; otherwise, $z \prec_\sigma u$ and $uz \notin E$ would contradict $z \prec_\tau u$ by lemma 34. There must be a vertex x separating u and z; $v \prec_\tau x$ and $xu \in E$. Obviously, $xv \in E$ by lemma 34. Since u is contained in the leftmost box containing a neighbour of x when x is chosen by `min-LexBFS`, $xz \notin E(H)$. Therefore, (u, x, z, v) is a cycle of length 4 in H, and uv is unique chord in this cycle. ∎

We did not only show that (u, x, z, v) is a cycle in H but also in G. Parra [38] proved that for a co-comparability graph G and any minimal triangulation H of G, an edge in H that is not in G is always unique chord in a cycle of length 4 in G.

Theorem 37. *Let $G = (V, E)$ be a co-comparability graph and σ a co-comparability ordering for G. Then, $\tau =$ `min-LexBFS`*(G, σ) is a minimal interval elimination scheme for G.*

Proof: Every subordering σ' of σ is a co-comparability ordering for the subgraph of G induced by the vertices occuring in σ'. As already discussed in the proof of theorem 24, τ restricted to the vertices of a component of G can be generated by `min-LexBFS`* with σ as second parameter and, therefore, is a minimal interval elimination scheme for this component by proposition 36. Furthermore, there is no vertex of one component that appears in τ between two vertices of another component by BFS-properties of `min-LexBFS`.

Hence, there is a 1-1 correspondance between the components of G and $H = (G, \tau)$ such that each component of H is a minimal triangulation of its corresponding component of G. H is a minimal triangulation of G. ∎

Corollary 38. *There is a linear-time algorithm that computes a minimal triangulation of a co-comparability graph.*

Proof: In linear time, we can compute a co-comparability ordering for G [32]. Applying theorem 37 and lemma 5, we obtain in overall linear time an interval model representation of a minimal triangulation of G. ∎

At the end of section 2, we discussed some properties of the graph G depicted in figure 1. We observed that it has a minimal triangulation that is not equal to (G, σ) for any BFS-ordering σ for G. Since G is a co-comparability graph, there is a co-comparability graph and a minimal triangulation of it that cannot be computed by our algorithm presented in corollary 38.

Our algorithms for computing minimal triangulations have a nice application to the Bandwidth problem. Let $G = (V, E)$ be a graph. A *layout* L of G is a 1-1 mapping from V to the set $\{1, \ldots, n\}$. The *width* $b(G, L)$ of layout L is the maximum over the set $\{L(u) - L(v) : uv \in E\} \cup \{0\}$. The *bandwidth* $bw(G)$ of G is the minimum width of a layout of G, i.e.

$$bw(G) = \min\{b(G, L) : L \text{ is layout of } G\}.$$

The decision problem Bandwidth $= \{(G, k) : bw(G) \leq k\}$ is NP-complete [37], even when restricted to the class of trees with maximum degree at most 3 [18]. Kloks, Kratsch, and Müller examined the Bandwidth problem for AT-free graphs and subclasses of AT-free graphs [27]. They proved that Bandwidth remains NP-complete on co-bipartite graphs. Since this class is a subclass of co-comparability graphs, the Bandwidth problem on co-comparability graphs is NP-complete. They also showed the following interesting result which links minimal triangulations of AT-free graphs, that are interval graphs by theorem 3, and the bandwidth problem.

Theorem 39. **[27]** *Let $G = (V, E)$ be an AT-free graph, and let H be a minimal triangulation of G. Then, $bw(G) \leq bw(H) \leq 2 \cdot bw(G)$.*

In [43], a polynomial-time algorithm for computing the bandwidth of an interval graph is presented, which is stated below.

Theorem 40. **[43]** *There is an algorithm that computes the bandwidth of a given interval graph G in time $\mathcal{O}(n \cdot \log^2 n)$, if G is given in form of an interval model.*

Connecting these results with our minimal triangulation computation algorithm for co-comparability graphs this provides an algorithm that approximates the bandwidth of a co-comparability graph with worst case performance ratio 2.

Theorem 41. *There is an algorithm that approximates in time $\mathcal{O}(n \cdot \log^2 n + m)$ the bandwidth of a co-comparability graph with worst case performance ratio 2.*

Proof: Let $G = (V, E)$ be a co-comparability graph. In linear time, we compute a minimal triangulation H of G that is given in form of an interval model (corollary 38). Applying the algorithm of theorem 40 to the interval model yields the bandwidth of H. By theorem 39, $bw(H) \leq 2 \cdot bw(G)$. ∎

In [27], a linear-time algorithm is given that approximates the bandwidth of co-comparability graphs with worst case performance ratio 3. If the number of edges of the input graph is of order at least $n \cdot \log^2 n$, our algorithm is an improvement of the former result.

For AT-free claw-free graphs, we can do even better. It was shown in [25] that the bandwidth of a proper interval graph is one less than the clique size of it, where the clique size means the number of vertices in the largest clique. Given the interval model this can be computed in time $\mathcal{O}(n)$ ([19]). Together with theorems 32 and 39, we obtain the following result which improves a previous algorithm by Fomin and Golovach [15].

Theorem 42. *There is a linear-time algorithm that approximates the bandwidth of an AT-free claw-free graph with worst case performance ratio 2.*

8. Conclusions. In this paper, we introduced a new algorithm `min-LexBFS` that generates an ordering of the vertex set of the input graph. A version of it for co-biartite graphs has already been presented in [33]. We studied properties of min-LexBFS-orderings and showed that the first vertex of such an ordering always belongs to a moplex. The first interesting application of `min-LexBFS` was a linear-time certifying recognition algorithm for proper interval graphs, a subclass of interval graphs. Further on, we developped linear-time algorithms to compute minimal triangulations of AT-free claw-free graphs and co-comparability graphs using `min-LexBFS`. Also in [33], related algorithms for both graph classes were presented. In case of AT-free claw-free graphs, we were able to prove that every minimal triangulation of such a graph can be computed by the presented algorithm, whereas there is a minimal triangulation of a co-comparability graph that cannot be computed by the given algorithm for co-comparability graph.

The most challenging open problem in this surrounding is the development of an algorithm for computing a minimal triangulation of an AT-free graph that runs faster than the currently known best algorithm with worst case time bound $\mathcal{O}(nm)$. Another direction of work is connected to `min-LexBFS` itself. We already observed strong parallels between `min-LexBFS` and `LexBFS` although generated orderings do not have to be equal. For instance, a min-LexBFS-ordering for a chordal graph is not always a simplical elimination scheme. How do both algorithms behave on special graph classes?

Acknowledgements. I am grateful to D. Kratsch, A. Berry, and D. Corneil for their interest in aspects of this work and for hints, remarks, and discussions. Furthermore, I want to thank an anonymous referee of WG2002 who initiated an improvement and extension of an earlier version.

9. Bibliography.

[1] A. Berry, *A wide-range efficient algorithm for minimal triangulation*, Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'99, pp., 1999.

[2] A. Berry, J.R.S. Blair, P. Heggernes, *Maximum Cardinality Search for Computing Minimal Triangulations*, presented at the 28th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2002), Cesky Krumlov, Czech Republic, June 2002.

[3] A. Berry, J.-P. Bordat, *Separability generalizes Dirac's theorem*, Discrete Applied Mathematics 84, pp. 43–53, 1998.

[4] J.R.S. Blair, P. Heggernes, J.A. Telle, *A Practical Algorithm for Making Filled Graphs Minimal*, Theoretical Computer Science 250, pp. 125–141, 2001.

[5] A. Brandstädt, V.B. Le, J.P. Spinrad, *Graph classes: a survey*, SIAM monographs on discrete mathematics and applications, 1999.

[6] D.G. Corneil, *A Simple 3-sweep LBFS Algorithm for the Recognition of Unit Interval Graphs*, to appear.

[7] D.G. Corneil, H. Kim, S. Natarajan, St. Olariu, A.P. Sprague, *Simple linear time recognition of unit interval graphs*, Information Processing Letters 55, pp. 99–104, 1995.

[8] D.G. Corneil, St. Olariu, L. Stewart, *Asteroidal triple-free graphs*, SIAM Journal on Discrete Mathematics 10, No. 3, pp. 399–430, 1997.

[9] D.G. Corneil, St. Olariu, L. Stewart, *The ultimate interval graph recognition algorithm? (Extended abstract)*, Proceedings of the Nineth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'98, pp. 175–180, 1998.

[10] D.G. Corneil, St. Olariu, L. Stewart, *Linear time algorithms for dominating pairs in asteroidal triple-free graphs*, SIAM Journal on Computing 28, No. 4, pp. 1284–1297, 1999.

[11] D.G. Corneil, St. Olariu, L. Stewart, *LBFS orderings and cocomparability graphs*, Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'99, pp. 883–884, 1999.

[12] E. Dahlhaus, *Minimal Elimination Ordering Inside a Given Chordal Graph*, Proceedings of the 23rd International Workshop on Graph-Theoretic Concepts in Computer Science, WG'97, Berlin, Lecture Notes in Computer Science 1335, pp. 132–143, Springer-Verlag, 1997.

[13] X. Deng, P. Hell, J. Huang, *Linear-time representation algorithm for proper circular-arc graphs and proper interval graphs*, SIAM Journal on Computing 25, pp. 390–403, 1996.

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

[15] F.V. Fomin, P.A. Golovach, *Interval degree and bandwidth of a graph*, To appear in: Discrete Applied Mathematics.

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

[17] T. Gallai, *Transitiv orientierbare Graphen*, Acta Mathematica Academiae Scientiarum Hungaricae 18, pp. 25–66, 1967.

[18] M.R. Garey, R.L. Graham, D.S. Johnson, D.E. Knuth, *Complexity results for Bandwidth minimization*, SIAM Journal on Applied Mathematics 34, No. 3, pp. 477–495, 1978.

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

[20] M.Ch. Golumbic, Cl. Monma, W.T. Trotter, Jr., *Tolerance Graphs*, Discrete Applied Mathematics 9, pp. 157–170, 1984.

[21] M. Habib, R. McConnell, Chr. Paul, L. Viennot, *Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing*, Theoretical Computer Science 234, pp. 59–84, 2000.

[22] P. Heggernes, Y. Villanger, *Efficient Implementation of a Minimal Triangulation Algorithm*, presented at the 10th European Symposium on Algorithms (ESA 2002), Rome, Italy, September 2002.

[23] H. Hempel, D. Kratsch, *On claw-free asteroidal triple-free graphs*, Discrete Applied Mathematics 121, pp. 155–180, 2002.

[24] C.M. Herrera de Figueiredo, J. Meidanis, C. Picinin de Mello, *A linear-time algorithm for proper interval graph recognition*, Information Processing Letters 56, pp. 179–184, 1995.

[25] H. Kaplan, R. Shamir, *Pathwidth, Bandwidth, and Completion Problems to Proper Interval Graphs with Small Cliques*, SIAM Journal on Computing 25, No. 3, pp. 540–561, 1996.

[26] T. Kloks, D. Kratsch, H. Müller, *Approximating the Bandwidth for Asteroidal Triple-Free Graphs*, Lecture Notes in Computer Science 979, ESA'95, pp. 434–447, 1995.

[27] T. Kloks, D. Kratsch, H. Müller, *Approximating the bandwidth for AT-free graphs*, Journal of Algorithms 32, pp. 41–57, 1999.

[28] D. Kratsch, R.M. McConnell, K. Mehlhorn, J.P. Spinrad, *Certifying Algorithms for Recognizing Interval Graphs and Permutation Graphs*, To be presented at SODA 2003.

[29] D. Kratsch, L. Stewart, *Domination on cocomparability graphs*, SIAM Journal on Discrete Mathematics 6, No. 3, pp. 400–417, 1993.

[30] C.G. Lekkerkerker, J.Ch. Boland, *Representation of finite graphs by a set of intervals on the real line*, Fundamenta Mathematicae 51, pp. 45–64, 1962.

[31] P.J. Looges, St. Olariu, *Optimal greedy algorithms for indifference graphs*, Computers & Mathematics with Applications 25, pp. 15–25, 1993.

[32] R.M. McConnell, J.P. Spinrad, *Modular decomposition and transitive orientation*, Discrete Mathematics 201, No. 1–3, pp. 189–241, 1999.

[33] D. Meister, *Minimale Triangulationen AT-freier Graphen*, Diploma thesis, Friedrich-Schiller-Universität Jena, Germany, 2000.

[34] R.H. Möhring, *Triangulating graphs without asteroidal triples*, Discrete Applied Mathematics 64, No. 3, pp. 281–287, 1996.

[35] St. Olariu, *An optimal greedy heuristic to color interval graphs*, Information Processing Letters 37, pp. 21–25, 1991.

[36] R. Paige, R.E. Tarjan, *Three partition refinement algorithms*, SIAM Journal on Computing 16, No. 6, pp. 973–989, 1987.

[37] Ch.H. Papadimitriou, *The NP-Completeness of the Bandwidth Minimization Problem*, Computing 16, pp. 263–270, 1976.

[38] A. Parra, *Structural and Algorithmic Aspects of Chordal Graph Embeddings*, Dissertation, 1996, Technische Universität Berlin.

[39] A. Parra, P. Scheffler, *Characterizations and algorithmic applications of chordal graph embeddings*, Discrete Applied Mathematics 79, pp. 171–188, 1997.

[40] F.S. Roberts, *Indifference graphs*, in: F. Harary (Ed.), Proof techniques in graph theory, pp. 139–146, Academic Press, New York, 1969.

[41] D.J. Rose, *Triangulated Graphs and the Elimination Process*, Journal of Mathematical Analysis and Applications 32, pp. 597–609, 1970.

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

[43] A.P. Sprague, *An $\mathcal{O}(n \log n)$ Algorithm for Bandwidth of Interval Graphs*, SIAM Journal on Discrete Mathematics 7, No. 2, pp. 213–220, 1994.

[44] M. Yannakakis, *Computing the minimum fill-in is NP-complete*, SIAM Journal on Algebraic Discrete Methods 2, pp. 77–79, 1981.