REPORTS
IN
INFORMATICS

ISSN 0333-3590

Determining treelength is hard

Daniel Lokshtanov

REPORT NO 333 **October 2006**

Department of Informatics

UNIVERSITY OF BERGEN

Bergen, Norway

This report has URL http://www.ii.uib.no/publikasjoner/texrap/pdf/2006-333.pdf

Reports in Informatics from Department of Informatics, University of Bergen, Norway, is available at http://www.ii.uib.no/publikasjoner/texrap/.

Requests for paper copies of this report can be sent to:

Department of Informatics, University of Bergen, Høyteknologisenteret,
P.O. Box 7800, N-5020 Bergen, Norway

Determining treelength is hard

Daniel Lokshtanov*

Abstract

In this article we resolve the computational complexity of determining the *treelength* of a graph, thereby solving an open problem of Dourusboure and Gavoille, who introduced this parameter, and asked to determine the complexity of recognizing graphs of bounded treelength [7]. The treelength of a graph G deals with the *length* of a tree-decomposition. The length of a tree decomposition of G is the largest distance in G between any two vertices appearing in the same tree node. The treelength of G is the minimum length of a tree decomposition of G. While recognizing graphs with treelength 1 is easily seen as equivalent to recognizing chordal graphs, something that can be done in linear time, the computational complexity of recognizing graphs with treelength 2 was unknown until this result. We show that the problem of determining whether a given graph has treelength at most k is NP-complete for every fixed $k \geq 2$.

1 Introduction

Treelength is a graph parameter proposed by Dourisboure and Gavoille [7] that measures how close a graph is to being chordal. The treelength of G is defined using tree decompositions of G. Graphs of treelength k are the graphs that have a tree decomposition where the distance in G between any pair of nodes that appear in the same bag of the tree decomposition is at most k. As chordal graphs are exactly those graphs that have a tree decomposition where every bag is a clique [13, 4, 10], we can see that treelength generalizes this charecterization.

There are several reasons for why it is interesting to study this parameter. For example, Dourisboure et. al. show that graphs with bounded treelength have sparse additive spanners [6]. Dourisboure also shows that graphs of bounded treelength admit compact routing schemes [5]. One should also note that many graph classes with unbounded treewidth have bounded treelength, such as chordal, interval, split, AT-free, and permutation graphs [7].

In this paper, we show that recognizing graphs with treelength bounded by a fixed constant $k \geq 2$ is NP-complete. The problem of settling the complexity of recognizing graphs of bounded treewidth was first posed as an open problem by Doursiboure and Gavoille, and remained open until this result [7]. Our result is somewhat surprising, because by bounding the treelength of G we put heavy restrictions on G's distance matrix. Another indication that this problem might be polynomial for fixed k was that the treelength of a graph is fairly easy to approximate within a factor of 3 [7]. In comparison, the best known approximation algorithm for treewidth has an approximation factor of $O(\log k)$ [1, 3]. As Bodlaender showed, recognizing graps with treewidth bounded by a constant can be done in linear time [2]. Since the above observation about approximability might indicate that determining treelength is "easier" than determining treewidth, one could arrive at the conclusion that recognizing graphs with treelength bounded by a constant should be polynomial. However, there are also strong arguments against this intuition. For instance, graphs of bounded treelength are just bounded diameter graphs that have been glued together in a certain way. Thus, when trying to show that a graph indeed has small treelength one would have to decompose the graph into components of small diameter and show how these components are glued together to form the graph. As the class of bounded diameter graphs is very rich, one would have a myriad of candidates to be such components, making it hard to pick out the optimal ones. This intuition is confirmed when we prove the

*Department of Informatics, University of Bergen, N-5020 Bergen, Norway. Email: daniel.lokshtanov@uib.no.

hardness of recognizing graphs of bounded treelength because the instances we reduce to all have bounded diameter.

In the next section we will give some notation and preliminary results. Next, we present a proof that determining whether the treelength of a weighted graph is less than or equal to k is NP-hard for every fixed $k \geq 2$. Following this, we reduce the problem of recognizing weighted graphs with treelength bounded by k to the problem of recognizing unweighted graphs with the treelength bounded by the same constant k, thereby completing the hardness proof. Finally we also consider the complexity of approximating treelength, and propose a fast exact algorithm to determine the parameter.

2 Notation, terminology and preliminaries

For a graph $G = (V, E)$ let $w : E \to \mathbb{N}$ be a *weight function* on the edges. The *length* of a path with respect to a weight function w is the sum of the weights of its edges. The *distance* $d_w(u, v)$ between two vertices is the length of the shortest path with respect to w. Whenever no weight function is specified the unit weight function $w(e) = 1$ for all $e \in E$ is used. G to the *power* of k with respect to the weight function w is $G^k_w = (V, \{uv : d_w(u, v) \leq k\})$. A weight function w is *metric* if it satisfies a generalization of the triangle inequality, that is, if $w((u, v)) = d_w(u, v)$ for every edge (u, v).

A *tree decomposition* of a graph $G = (V, E)$ is a pair (S, T) consisting of a set $S = \{X_i : i \in I\}$ of *bags* and a tree $T = (I, M)$ so that each bag $X_i \in S$ is a subset of V and the following conditions hold:

- $\bigcup_{i \in I} X_i = V$
- For every edge (u, v) in E, there is a bag X_i in S so that $u \in X_i$ and $v \in X_i$
- For every vertex v in V, the set $\{i \in I : v \in X_i\}$ induces a connected subtree of T

The *length* of a bag is the maximum distance in G between any pair of vertices in the bag. The *length* of a tree-decomposition is the maximum length of any bag. The *treelength* of G with weight function w is the minimum length of a tree-decomposition of G, and is denoted by $tl_w(G)$. A *shortest* tree decomposition is a tree decomposition having minimum length. We will always assume that all weight functions are metric. This can be justified by the fact that if w is not metric, we easily can make a new metric weight function w' by letting $w'((u, v)) = d_w(u, v)$ for every edge (u, v) and observe that $tl'_w(G) = tl_w(G)$.

The *neighbourhood* of a vertex v is denoted $N(v)$ and is the vertex set $\{u : (u, v) \in E\}$. When S is a subset of V, $G[S] = (S, E \cap \{(u, v) : u \in S, v \in S\})$ is the subgraph *induced* by S. G is *complete* if (u, v) is an edge of G for every pair $\{u, v\}$ of distinct vertices in G. A *clique* in G is a set S of vertices in G so that $G[S]$ is complete.

For two graphs $G = (V, E)$ and $G' = (V, E')$, $G \subseteq G'$ means that $E \subseteq E'$. A *graph class* is a set of graphs that is closed under graph isomorphism. For a graph class Π, G is a Π-graph if $G \in \Pi$. G' is a Π-*sandwich* between G and G'' if G' is a Π-graph and $G \subseteq G' \subseteq G''$ [11]. A graph class Π is *hereditary* if every induced subgraph of a Π-graph is a Π-graph.

A graph is *chordal* if it contains no induced cycle of length at least 4. Thus one should note that the class of chordal graphs is hereditary. A vertex v is *simplicial* if the neighbourhood of v is a clique. A vertex v is *universal* if $V = \{v\} \cup N(v)$. An ordering of the vertices of G into $\{v_1, v_2, \ldots, v_n\}$ is a *perfect elimination ordering* if for every i, v_i is simplicial in $G[\{v_j : j \geq i\}]$. A *clique tree* of G is a tree decomposition of G so that every bag is a clique. [12]

Theorem 2.1 *The following are equivalent:*

- *G is chordal*
- *G has a clique tree [13, 4, 10]*
- *G has a perfect elimination ordering. [9]*

For more characterizations of chordal graphs and the history of this graph class, refer to the survey by Heggernes [12]. Following Theorem 2.1 it is easy to see that if v is simplicial then G is chordal if and only if $G \setminus v$ is chordal. Universal vertices share this property, as has been observed by several authors before.

Observation 2.2 (Folklore) *If v is universal then G is chordal if and only if $G \setminus v$ is chordal*

Proof. If G is chordal then $G \setminus v$ is chordal because the class of chordal graphs is hereditary. Now suppose $G \setminus v$ is chordal. Consider a perfect elimination ordering of $G \setminus v$ appended by v. This is clearly a perfect elimination ordering of G, hence G is chordal. ∎

We now define the problem that we are going to show is NP-complete.

k-TREELENGTH
INSTANCE: A graph G
PARAMETER: An integer $k \geq 2$
QUESTION: Is $tl(G) \leq k$?

Finally, we define and the problem we will reduce from.

CHORDAL SANDWICH PROBLEM [11]
INSTANCE: Two graphs G_1 and G_2 with $G_1 \subseteq G_2$
QUESTION: Is there a chordal sandwich between G_1 and G_2?

3 Weighted k-Treelength is NP-Complete

In this section we are going to show that determining whether the treelength of a given weighted graph is at most k is NP-complete for every fixed $k \geq 2$. In the next section we will conclude the hardness proof for unweighted graphs by showing how one from a weighted graph G in polynomial time can construct an unweighted graph G' with the property that $tl_w(G) \leq k$ if and only if $tl(G') \leq k$.

WEIGHTED k-TREELENGTH
INSTANCE: A graph G with weight function w
PARAMETER: An integer $k \geq 2$
QUESTION: Is $tl_w(G) \leq k$?

Observation 3.1 *For a graph $G = (V, E)$, $tl_w(G) \leq k$ if and only if there exists a chordal sandwich G' between G and G_w^k.*

Proof. Suppose $tl_w(G) \leq k$. Consider a shortest tree decomposition (S, T) of G, and construct the graph $G' = (V, \{(u, v) : \exists i \ u \in X_i, v \in X_i\})$. $G \subseteq G'$ is trivial, $G' \subseteq G_w^k$ holds because the length of the tree decomposition is at most k, and G' is chordal because (S, T) is a clique tree of G'. In the other direction, let G' be a chordal sandwich between G and G_w^k. Consider a clique tree (S, T) of G'. This is a tree decomposition of G, and the length of this decomposition is at most k, as $u \in X_i$ and $v \in X_i$ implies $(u, v) \in E(G') \subseteq E(G_w^k)$. ∎

This yields the following corollary

Corollary 3.2 *$tl(G) = 1$ if and only if G is chordal.*

From Observation 3.1, it follows that determining the treelength of a given graph in fact is a special case of *The Chordal Sandwich* problem defined above. In a study of sandwich problems [11], Golumbic et. al. point out that as a consequence of the hardness of Triangulating Colored Graphs, the Chordal Sandwich problem is NP-Complete. Thus, in order to prove that Weighted k-Treelength is indeed hard, we only need to reduce the Chordal Sandwich problem to a special case of itself, namely the one where $G_2 = {G_1}_w^k$ for some weight function w.

We will reduce in the following way. On input $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ with $G_1 \subseteq G_2$ to the Chordal Sandwich problem, let $E_D = E_2 \setminus E_1$. We construct a new graph G by taking a copy of G_1, adding a new vertex c_{uv} for every edge (u, v) in E_D and making this vertex adjacent to all other vertices of G. We denote the set of added vertices by C, as C is a clique of universal vertices. The weight function is simple, $w(c_{uv}, u) = w(c_{uv}, v) = \lfloor k/2 \rfloor$ for every c_{uv} and $w(e) = k$ for all other edges.

Lemma 3.3 *Let G, G_1 and G_2 be as described above. Then $tl_w(G) \leq k$ if and only if there is a chordal sandwich G' between G_1 and G_2.*

Proof. Observe that a graph $G' \supseteq G$ is chordal if and only if $G'[V_1]$ is chordal since every vertex in C is universal. Also, notice that for every pair u,v of vertices in V_1, $d_w(u, v) \leq k$ if and only if (u, v) is an edge of G_2. Thus it follows that $G_2 = G_w^k[V_1]$. Hence, by Observation 3.1, $tl_w(G) \leq k$ if and only if there is a chordal sandwich G' between G and G_w^k. By the discussion above, this is true if and only if there is a chordal sandwich between $G[V_1] = G_1$ and $G_2 = G_w^k[V_1]$. ∎

Corollary 3.4 *Weighted k-Treelength is NP-complete for every $k \geq 2$.*

Proof. By Lemma 3.3 determening whether a given weighted graph G has $tl_w(G) \leq k$ is NP-hard for every $k \geq 2$. By Observation 3.1 this problem is polynomial time reducible to the Chordal Sandwich Problem, thus it is in NP. ∎

4 k-Treelength is NP-Complete

We will now show how one from a weighted graph G in polynomial time can construct an unweighted graph G'' with the property that $tl_w(G) \leq k$ if and only if $tl(G'') \leq k$. We do this in two steps. First we show how to construct a graph G' and weight function w' from G and w so that $tl_w(G) \leq k$ if and only if $tl_{w'}(G') \leq k$ and $w'(e) = 1$ or $w'(e) = k$ for every edge e in G'. In the second step we show how G'' can be constructed from G' and w'. Both steps are done in an inductive way. Obviously, if G has an edge of weight larger than k then $tl_w(G) > k$. We will therefore assume that $w(e) \leq k$ for all edges e. For an edge (u, v), let $G(u, v) = (V \cup \{r, q\}, (E \setminus (u, v)) \cup \{(u, r),(r, v),(u, q),(q, v)\})$. That is, we build $G(u, v)$ from G by removing the edge (u, v), adding two new vertices r and q and making both of them adjacent to u and v. Let $w_{u,v,k}$ be a weight function of $G(u, v)$ so that $w_{(u,v,k)}(e) = w(e)$ if $e \in E(G) \cap E(G(u, v))$, $w_{(u,v,k)}((u, r)) = w_{(u,v,k)}((r, v)) = k$, $w_{(u,v,k)}((u, q)) = w(u, v) - 1$, and $w_{(u,v,k)}((q, v)) = 1$. Observe that if $w((u, v)) > 1$ then $w_{(u,v,k)}$ is properly defined.

Lemma 4.1 *Given a graph G, an edge (u, v), and a weight function w with $w((u, v)) > 1$, there is a chordal sandwich between G and G_w^k if and only if there is a chordal sandwich between $G(u, v)$ and $G(u, v)^k_{w_{(u,v,k)}}$.*

Proof. Suppose there is a chordal sandwich $\hat{G}_{(u,v)}$ between $G(u, v)$ and $G(u, v)^k_{w_{(u,v,k)}}$. Then the edge (u, v) must be in $E(\hat{G}_{(u,v)})$ and thus $\hat{G}_{(u,v)} \setminus \{r, q\}$, where r and q are the vertices that were added to G (u, v) to obtain $G(u, v)$, is a chordal sandwich between G and G_w^k. In the other direction, suppose there is a chordal sandwich $\hat{G}$ between G and and G_w^k. Then $\hat{G}' = (V(\hat{G}) \cup \{r, q\}, E(\hat{G}) \cup \{(u, r), (r, v), (u, q), (q, v)\})$ is a chordal sandwich between $G(u, v)$ and $G(u, v)^k_{w_{(u,v,k)}}$ because the r and q are simplicial nodes in $\hat{G}'$. ∎

Now, the idea is that the graph $G(u, v)$ with weight function $w_{(u,v,k)}$ is somewhat closer to not having any edges with weight between 2 and $k - 1$. With an appropriate choice of measure, it is easy to show that this is indeed the case. The measure we will use will essentially be the sum of the weigts of all edges that have edge weights between 2 and $k - 1$. In the follwing discussion, let $W_w(G) = \sum_{e \in E, w(e) < k} w(e) - 1$. Observe that if $1 < w(u, v) < k$ then $W_{w_{(u,v,k)}}(G(u, v)) = W_w(G) - 1$, and that if $W_w(G) = 0$ then $w(e) = 1$ or $w(e) = k$ for every edge $e \in E$.

Lemma 4.2 *For a graph G with weight function w, we can construct in polynomial time a graph G' with weight function w' so that $|V(G')| = |V(G)| + 2W_w(G)$, and $tl_w(G) \leq k$ if and only if $tl_{w'}(G') \leq k$.*

Proof. We prove by induction on $W_w(G)$. If $W_w(G) = 0$ we know that $w(e) = 1$ or $w(e) = k$ for every edge e. Now, suppose the statement of the lemma holds for all graphs with $W_w(G) < t$ for some t and consider a graph G with weight function w so that $W_w(G) = t > 0$. Then, let (u, v) be an edge so that $1 < w((u, v)) < k$. By Lemma 4.1, $tl_w(G) \leq k$ if and only if $tl_{w_{(u,v,k)}}(G(u, v)) \leq k$. Now, $W_{w_{(u,v,k)}}(G(u, v)) = W_w(G)-1$. Thus, by the induction assumption, we can in polynomial time construct a graph G' with weight function w' that satisfies $tl_w(G) \leq k \iff tl_{w_{(u,v,k)}}(G(u, v)) \iff tl_{w'}(G') \leq k$ with $|V(G')| = |V(G(u, v))|+2W_{w_{(u,v,k)}}(G(u, v)) = |V(G)|+2+2(W_w(G)-1) = |V(G)|+2W_w(G)$. ∎

The idea of the above proof is that we can use edges of weight 1 and k to emulate the behaviour of edges with other weights. The method we now will use to prove the hardness of unweighted treelength will be similar - we are going to show that weight k edges can be emulated using only edges with weight 1. In order to do this, we are going to use the following lemma by Dourisboure et. al. concerning the treelength of cycles.

Lemma 4.3 *[7] The treelength of a cycle on k vertices is $\lceil \frac{k}{3} \rceil$.*

For an edge $(u, v) \in E$, we construct the graph $G[u, v, k]$ in the following way: We replace the edge (u, v) by three paths on $2k - 1$, $2k - 1$ and $k - 1$ vertices respectively. Construct these paths $P_a = \{a_1, a_2, \ldots, a_{2k-1}\}$, $P_b = \{b_1, b_2, \ldots, b_{2k-1}\}$ and $P_c = \{c_1, c_2, \ldots, c_{k-1}\}$ using new vertices. Take a copy of G, remove the edge (u, v) and add edges from u to a_1, b_1 and c_1, and from v to a_{2k-1}, b_{2k-1} and c_{k-1}. For a weight function w of G, $w_{[u,v,k]}$ will be a weight function of $G[u, v, k]$ so that $w_{[u,v,k]}(e) = w(e)$ if $e \in E(G)$ and $w_{[u,v,k]} = 1$ otherwise.

Lemma 4.4 *Given G, weight function w and an edge $(u, v) \in E$ with $w(u, v) = k$, $tl_w(G) \leq k$ if and only if $tl_{w_{[u,v,k]}}(G[u, v, k]) \leq k$*

Proof. Suppose there is a chordal sandwich $\hat{G}$ between G and G^k_w. We build $\hat{G}'$ from G by taking a copy of $\hat{G}$, adding three new paths $P_a = \{a_1, a_2, \ldots, a_{2k-1}\}$, $P_b = \{b_1, b_2, \ldots, b_{2k-1}\}$ and $P_c = \{c_1, c_2, \ldots, c_{k-1}\}$ and the edge sets $\{(u, a_i) : i \leq k\}$, $\{(u, b_i) : i \leq k\}$, $\{(u, c_i) : i \leq \lfloor \frac{k}{2} \rfloor\}$, $\{(v, a_i) : i \geq k\}$, $\{(v, b_i) : i \geq k\}$, $\{(v, c_i) : i \geq \lfloor \frac{k}{2} \rfloor\}$. We see that $\hat{G}'$ is chordal because $\{a_1, a_2 \ldots a_{k-1}, a_{2k-1}, a_{2k-2}, \ldots a_k, b_1, b_2 \ldots b_{k-1}, b_{2k-1}, b_{2k-2}, \ldots b_k, c_1, c_2, \ldots c_{\lfloor \frac{k}{2} \rfloor - 1}, c_{k-1}, c_{k-2}, \ldots c_{\lfloor \frac{k}{2} \rfloor}\}$ followed by a perfect elimination ordering of $\hat{G}$ is a perfect elimination ordering of $\hat{G}'$. Also, $\hat{G}' \subseteq G[u, v, k]^k_{w_{[u,v,k]}}$. Thus $\hat{G}'$ is a chordal sandwich between $G[u, v, k]$ and $G[u, v, k]^k_{w_{[u,v,k]}}$. In the other direction, let $\hat{G}_{[u,v]}$ be a chordal sandwich between $G[u, v, k]$ and $G[u, v, k]^k_{w_{[u,v,k]}}$. It is sufficient to show that $(u, v) \in E(\hat{G}_{[u,v]})$ because then $\hat{G}_{[u,v]}[V(G)]$ is a chordal sandwich between G and G^k_w. Consider the set $V_s = \{u, v\} \cup V(P_a) \cup V(P_b)$, and let C be the subgraph of $G[u, v, k]$ induced by V_s. Now, observe that $E(G[u, v, k]^k_{w_{[u,v,k]}}[S]) = E(C^k) \cup \{(u, v)\}$. Suppose for contradiction that (u, v) is not an edge of $\hat{G}_{[u,v]}$. Then we know that $\hat{G}_{[u,v]}[V_s]$ is a chordal sandwich between C and C^k implying that $tl(C) \leq k$. This contradicts Lemma 4.3 because C is a cycle on $4k$ vertices. ∎

Lemma 4.4 gives us a way to emulate edges of weight k using only edges of weight 1. For a graph G with weight function w, let $W_w[G] = |\{e \in E(G) : w(e) = k\}|$. Notice that if $w((u, v)) = k$ then $W_w[G] = W_{w_{[u,v,k]}}[G[u, v, k]] + 1$.

Lemma 4.5 *For every graph G with weight function w satisfying $w(e) = 1$ or $w(e) = k$ for every edge, we can construct in polynomial time a graph G' with $W_w[G](5k - 3) + |V(G)|$ vertices and satisfying $tl_w(G) \leq k$ if and only if $tl(G') \leq k$.*

Proof. We use induction in $W_w[G]$. If $W_w[G] = 0$ the lemma follows immediately. Now, assume the result holds for $W_w[G] < t$ for some $t > 0$. Consider a graph G with weight function w so that $W_w[G] = t$. By Lemma 4.4 $tl_w(G) \leq k$ if and only if $tl_{w_{[u,v,k]}}(G[u, v, k]) \leq k$. By the inductive hypothesis we can construct in polynomial time a graph G' with $W_{w_{[u,v,k]}}[G[u, v, k]](5k - 3) + |V(G[u, v, k])| + 5k - 3 = W_w[G](5k - 3) + |V(G)|$ vertices and satisfying $tl_(G') \leq k \iff tl_{w_{[u,v,k]}}(G[u, v, k]) \leq k \iff tl_w(G) \leq k$ ∎

Corollary 4.6 *For a graph G and weight function w, we can in polynomial time construct a graph G'' so that $tl_w(G) \leq k$ if and only if $tl(G'') \leq k$.*

Proof. By Lemma 4.2 we can in polynomial time construct a graph G' with weight function w' so that $tl_{w'}(G') \leq k \iff tl_w(G) \leq k$ and so that $w'(e) = 1$ or $w'(e) = k$ for every edge e in $E(G')$. By Lemma 4.5 we can from such a G' and w' construct in polynomial time a G'' so that $tl(G'') \leq k \iff tl_{w'}(G') \leq k \iff tl_w(G) \leq k$. ∎

Theorem 4.7 *Determining whether a graph G has $tl(G) \leq k$ is NP-complete for every fixed $k \geq 2$.*

Proof. By Corollary 4.6, k-Treelength is NP-hard. As it is a special case of Weighted k-Treelength it is also NP-complete. ∎

5 Conclusion

We have proved that it is NP-complete to recognize graphs with treelength bounded by a constant $k \geq 2$. Thus it would be interesting to see what attacks can be made on the problem in the fields of exact algorithms and approximation algorithms. It seems that a simple modification of the exact algorithm by Fomin et. al [8] for computing treewidth and minimum fill can be used to compute the treelength of a graph in time $O(1.8899^n)$. As for approximation algorithms, Dourisboure and Gavoille provide two 3-approximation algorithms for treelength in [7], and propose a heuristic that they conjecture is a 2-approximation algorithm. A 2-approximation for treelength would thus be of interest. One should note that the hardness proof 2-Treelength automatically implies that if $P \neq NP$ there is no polynomial time approximation algoritm for treelength with approximation ratio better than $\frac{3}{2}$. As the gap between the lower and the upper bound for the approximation ratio is fairly small, and both the lower and upper bounds stem from fairly simple arguments, one can hope for results that close this gap.

References

[1] E. Amir. Efficient approximation for triangulation of minimum treewidth. *Proceedings UAI 2001*, pages 7–15, 2001. 1

[2] H.L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25:1305–1317, 1996. 1

[3] V. Bouchitte, D. Kratsch, H. Muller, and I. Todinca. On treewidth approximations. *Discrete Applied Mathematics*, 136(2-3):183–196, 2004. 1

[4] P. Buneman. A characterization of rigid circuit graphs. *Discrete Mathematics*, 9:205–212, 1974. 1, 2.1

[5] Y. Dourisboure. Compact routing schemes for bounded tree-length graphs and for k-chordal graphs. *Proceedings DISC 2004, Lecture Notes in Computer Science*, 3274:365–378, 2004. 1

[6] Y. Dourisboure, F.F. Dragan, C. Gavoille, and C. Yan. Sparse additive spanners for bounded tree-length graphs. *To appear in Theoretical Computer Science*. 1

[7] Y. Dourisboure and C. Gavoille. Tree-decompositions with bags of small diameter. *To appear in Discrete Mathematics*. (document), 1, 4.3, 5

[8] F.V. Fomin, D. Kratsch, I. Todinca, and Y. Villanger. Exact algorithms for treewidth and minimum fill-in. *SIAM Journal on Computing, submitted*. 5

[9] D.R. Fulkerson and O.A Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15:835–855, 1965. 2.1

[10] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graps. *Journal of Combinatorial Theory B*, 16:47–56, 1974. 1, 2.1

[11] M.C. Golumbic, H. Kaplan, and R. Shamir. Graph sandwich problems. *J. Algorithms*, 19(3):449–473, 1995. 2, 2, 3

[12] P. Heggernes. Minimal triangulations of graphs: A survey. *Discrete Mathematics*, 306(3):297–317, 2006. 2, 2

[13] J. Walter. *Representation of rigid cycle graphs*. PhD thesis, 1972. 1, 2.1