UNIVERSITE D'ORLEANS

Faculté des Sciences

LIFO

Laboratoire d'Informatique Fondamentale d'Orléans
4, rue Léonard de Vinci, BP 6759
F-45067 Orléans Cedex 2
FRANCE

Rapport de Recherche

www : http://www.univ-orleans.fr/SCIENCES/LIFO/

On treewidth approximations

V. Bouchitté
LIP, École Normale Supérieure de Lyon
D. Kratsch
LITA - Université de Metz
H. Müller
School of Computing, University of Leeds
I. Todinca
LIFO, Université d'Orléans

Rapport N^o **2001-04**

On treewidth approximations

V. Bouchitté[1], D. Kratsch[2], H. Müller[3], and I. Todinca[4]

[1] LIP - Ecole normale supérieure de Lyon, 46 Allée d'Italie, 69364 Lyon cedex 07, France, Vincent.Bouchitte@ens-lyon.fr
[2] LITA - Université de Metz, Ile du Saulcy, 57045 Metz Cedex 01, France, kratsch@lita.univ-metz.fr
[3] School of Computing, University of Leeds, Leeds LS2 9JT, United Kingdom, hm@comp.leeds.ac.uk
[4] LIFO - Université d'Orléans, BP 6759, 45067 Orléans Cedex 2, France, Ioan.Todinca@lifo.univ-orleans.fr

October 17, 2001

Abstract. We introduce a natural heuristic for approximating the treewidth of graphs. We prove that this heuristic gives a constant factor approximation for the treewidth of graphs with bounded asteroidal number. Using a different technique, we give a $\mathcal{O}(\log \mathrm{OPT})$ approximation algorithm for the treewidth of arbitrary graphs.

1 Introduction

A graph is said to be *chordal* if each cycle with at least four vertices has a chord, that is an edge between two non-consecutive vertices of the cycle. A triangulation of a graph is a chordal supergraph having the same vertex set. The *treewidth* of a graph G is the minimum cliquesize over all possible triangulations of G, minus one.

The notion of treewidth has been intensively studied in the last years, mainly because many classical NP-hard problems become polynomial and even linear when restricted to graphs with small treewidth. These algorithms often need an optimal triangulation of the input graph. More precisely, given a graph G and a triangulation H of G, the running time of these algorithms is polynomial in the size of the graph and exponential in the cliquesize of the triangulation.

Computing the treewidth of arbitrary graphs is NP-hard. Nevertheless, the treewidth can be computed in polynomial time for several well-known classes of graphs, for example the chordal bipartite graphs, the circle and circular-arc graphs, and the permutation graphs. All these algorithms use the *minimal separators* of the graph and the fact that these classes of graphs have "few" minimal separators, in the sense that the number of the separators is polynomially bounded in the size of the graph. For a class of graphs having a polynomial number of minimal separators, the treewidth is computable in polynomial time [5, 6].

The existence of a polynomial time approximation algorithm which is no more than a constant times the optimal value is a question that still remains open. In [4], we gave a 2-approximation algorithm for the treewidth of AT-free graphs. Here we introduce a greedy heuristic for computing minimal triangulations of a graph. The heuristic tries to minimize the cliquesize of the triangulation. We show that our heuristic is a constant factor approximation for the treewidth of graphs with bounded asteroidal number. Since AT-free graphs are graphs with asteroidal number at most two, this generalizes the result of [4]. We prove that, unfortunately, the heuristic does not give a constant factor approximation for the treewidth of arbitrary graphs.

The best known result on the treewidth approximation of arbitrary graphs is a $\mathcal{O}(\log n)$ approximation algorithm by Bodlaender et al [3]. We modify here their algorithm in order to obtain a $\mathcal{O}(\log k)$ approximation factor, where k is the treewidth of the input graph. To our knowlege, this is the first approximation result for the treewidth problem, in which the approximation factor does not depend on the size of the input graph.

2 Preliminaries

Throughout this paper we consider simple, finite, undirected graphs.

A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(T, \mathcal{X})$, where $T = (I, F)$ is a tree and $\mathcal{X} = \{X_i \mid i \in I\}$ is a collection of subsets of V such that

1. $\bigcup_{i \in I} X_i = V$.
2. $\forall xy \in E$, there is an $i \in I$ such that X_i contains both x and y.
3. For each vertex $x \in V$, the set of nodes $\{i \in I \mid x \in X_i\}$ forms a connected subtree of T.

We will say that the set X_i is the *label* of the node i in T. The *width* of a tree-decomposition $(T = (I, F), \mathcal{X})$ is $\max_{i \in I} |X_i| - 1$. The treewidth of the graph G is the minimum width over all tree-decompositions of the graph.

It is well-known that a tree-decomposition of a graph G corresponds to a triangulation of G. We restate the definition of treewidth in terms of triangulations. A graph is said to be *chordal* or *triangulated* if each cycle with at least four vertices has a chord, that is an edge between two non-consecutive vertices of the cycle. Given an arbitrary graph $G = (V, E)$, a *triangulation* of G is a chordal graph $H = (V, F)$ such that $E \subseteq F$. We say that H is a *minimal triangulation* of G if no proper subgraph of H is a triangulation of G. The *treewidth* $\mathrm{tw}(H)$ of a chordal graph is its maximum cliquesize minus one. The treewidth of an arbitrary graph G is the minimum, over all triangulations H of G, of $\mathrm{tw}(H)$.

When computing the treewidth of G we can clearly restrict to minimal triangulations. The *minimal separators* play a crucial role in the characterization of the minimal triangulations of a graph.

A subset $S \subseteq V$ is an *a, b-separator* for two nonadjacent vertices $a, b \in V$ if the removal of S from the graph separates a and b in different connected components. S is a *minimal a, b-separator* if no proper subset of S separates a and b. We say that S is a *minimal separator* of G if there are two vertices a and b such that S is a minimal a, b separator. Notice that a minimal separator can be strictly included into another. We denote by Δ_G the set of all minimal separators of G.

Let G be a graph and S a set of vertices of G. We note $\mathcal{C}_G(S)$ the set of connected components of $G \backslash S$. A component $C \in \mathcal{C}_G(S)$ of $G \backslash S$ is a *full component associated to S* if every vertex of S is adjacent to some vertex in C.

Let S be a minimal separator of G. If $C \in \mathcal{C}_G(S)$, we say that $(S, C) = S \cup C$ is a *one-block* associated to S. A one-block (S, C) is called *full* if C is a full component associated to S. If (S, C) is a full one-block, then $S = N_G(C)$. If (S, C) is not full, then $S^* = N_G(C)$ is a minimal separator of G, strictly contained in S.

Let S be a minimal separator of G. We say that S *crosses* a set of vertices A if S separates two vertices $x, y \in A$ (i.e. S is an x, y-separator). We say that S *separates* two sets of vertices A and B if S separates each vertex of $A \backslash S$ from each vertex of $B \backslash S$.

Let S and T be two minimal separators. If S crosses T, we write $S \# T$. Otherwise, S and T are called *parallel*, denoted by $S \| T$. It is easy to prove that these relations are symmetric. Remark that S and T cross if and only if T intersects each full component associated to S. Conversely, S and T are parallel if and only if T is contained in some one-block (S, C_T) associated to S. In particular, if $T \subseteq S$, then S and T are parallel.

Let $S \in \Delta_G$ be a minimal separator. We denote by G_S the graph obtained from G by *completing* S, i.e. by adding an edge between every pair of non-adjacent vertices of S. If $\Gamma \subseteq \Delta_G$ is a set of separators of G, G_Γ is the graph obtained by completing all the separators of Γ. The results of [10], concluded in [12], establish a strong relation between the minimal triangulations of a graph and its minimal separators.

Theorem 1. *Let $\Gamma \in \Delta_G$ be a maximal set of pairwise parallel separators of G. Then $H = G_\Gamma$ is a minimal triangulation of G and $\Delta_H = \Gamma$.*

Let H be a minimal triangulation of a graph G. Then Δ_H is a maximal set of pairwise parallel separators of G and $H = G_{\Delta_H}$.

In other terms, every minimal triangulation of a graph G is obtained by considering a maximal set Γ of pairwise parallel separators of G and completing the separators of Γ. The minimal separators of the triangulation are exactly the elements of Γ.

3 The minimum cardinality separator strategy

By theorem 1, any minimal triangulation of G is obtained by completing a maximal set of pairwise parallel separators of G. The following proposition ([12]) says that we can complete these separators one by one.

Proposition 1. *Let Γ' be a set of pairwise parallel minimal separators of the graph G and let $H' = G_{\Gamma'}$. Then S is a minimal separator of H' if and only if S is a minimal separator of G, parallel in G to each $T \in \Gamma'$.*

This leads to the following algorithm computing minimal triangulation of graphs.

MinimalTriangulation
$H \leftarrow G$
`while` H is not chordal
  choose $S \in \Delta_H$ such that $H[S]$ is not a clique
  $H \leftarrow H_S$
`end_while`
`return` H

Not only this algorithms always produces a minimal triangulation of G, but any minimal triangulation of G can be produced by the algorithm, by choosing the appropriate minimal separators.

Our aim is to obtain a triangulation H of minimum cliquesize. Since the minimal separators S chosen in the `while` loop become cliques, a natural idea is to choose a minimal separator S of minimum cardinality. This gives the following algorithm, that we call the *minimum cardinality separator strategy*.

MCSep
$H \leftarrow G$
`while` H is not chordal
  choose **the smallest** $S \in \Delta_H$ such that $H[S]$ is not a clique
  $H \leftarrow H_S$
`end_while`
`return` H

4 Blocks

Given a graph H, we do not know how to compute a minimum size separator S of H such that $H[S]$ is not a clique. We present an alternative version of the minimum cardinality separator strategy for which we only need to compute a minimum cardinality separator of a graph. Finding a minimum size separator of a graph can be done in polynomial time by standard flow techniques [1].

Let us give first some further definitions, which are strongly related with the *blocking sets* and the *blocks* introduced in [7].

Definition 1. *Let G be a graph and $\mathcal{S} \subseteq \Delta_G$ a set of pairwise parallel separators such that for any $S \in \mathcal{S}$, there is a one-block $(S, C(S))$ containing all the elements of $\mathcal{S}$. Suppose that $\mathcal{S}$, ordered by inclusion, has no greatest element. We define the* piece between *the elements of $\mathcal{S}$ by*

$$P(\mathcal{S}) = \bigcap_{S \in \mathcal{S}} (S, C(S))$$

Notice that for any $S \in \mathcal{S}$ the one-block of S containing all the separators of $\mathcal{S}$ is unique: if $T \in \mathcal{S}$ is not included in S, there is a unique connected component of $G \backslash S$ containing $T \backslash S$.

Definition 2. *Let B be a vertex set of a graph G. We denote by $C_1, \ldots, C_p$ the connected components of $G \backslash B$ and by S_i the neighborhood of C_i. We will say that B is a* block *of G if the sets S_i are minimal separators of G and one of the following conditions holds:*

1. *$B = G$.*
2. *There is an $i \in [1, p]$ such that B is a one-block (S_i, C).*
3. *$B = P(S_1, \ldots, S_p)$.*

Definition 3. *Let B be a block, let $C_1, \ldots, C_p$ be the connected components of $G \backslash B$ and $S_i = N(C_i)$. We say the minimal separators $S_1, \ldots, S_p$* border *B. Let $\mathcal{S}(B)$ be the set of minimal separators bordering B and let $\mathcal{BS}(B)$ be the inclusion-maximal elements of $\mathcal{S}(B)$. Then $\mathcal{BS}(B)$ is called the* blocking set *of B.*

Proposition 2 ([7]). *Let B be a block of G. If B is a one-block (S, C), then the blocking set of B is $\{S\}$. If B is the piece between some minimal separators, then B is also the piece between the elements of its blocking set.*

Definition 4. *Let B be a block of G and let $\mathcal{S}(B)$ be the minimal separators bordering B. The graph $R(B) = G_{\mathcal{S}(B)}[B]$ obtained from B by completing each $S \in \mathcal{S}(B)$ into a clique is called the* realization *of B.*

Any minimal separator of the realisation of some block B is also a minimal separator of G:

Theorem 2 ([7]). *Let B be a block of G and S be a minimal separator of its realization $R(B)$. Then S is a minimal separator of G.*

Moreover, for each connected component C of $R(B) \backslash S$, $S \cup C$ is a block of G. Each minimal separator bordering $S \cup C$ is contained in S or belongs to $\mathcal{S}(B)$.

Conversely, a minimal separator S of G contained in some block B is either one of the separators bordering B, or it is a minimal separator of $R(B)$.

Theorem 3 ([7]). *Let B be a block of G and S be a minimal separator of G, contained in B. If S separates in G two vertices of B, then S is also a minimal separator of $R(B)$.*

5 Implementation of the heuristic

In [2], an algorithm is given for computing a minimal triangulation of a graph, by recursevely splitting blocks. We slightly modify it in order to abtain an implementation of our minimum cardinality separator strategy. The algorithm maintains a list of blocks, called lb. Initialy $lb = \{G\}$. At each step, we split a block B by choosing a minimum cardinality separator S of $R(B)$. Then B is replaced in the list of blocks by the smaller blocks $S \cup C_i$, where C_i are the connected components of $R(B)\backslash S$. The algorithm stops when the realizations of all blocks in lb are cliques. The output is the graph $H = G_{lb}$, obtained from G by completing each block of lb into a clique.

Here is the pseudo-code of the **MCSep** algorithm:

MCSep
`Input:` G
`Output:` a minimal triangulation of G
  $lb \leftarrow \{G\}$ / $*$ lb is the list of blocks $*$/
  `while` there is $B \in lb$ such that $R(B)$ is not a clique `do`
    $S \leftarrow$ a minimum cardinality separator of $R(B)$
    compute the connected components $C_1, \ldots C_p$ of $R(B)\backslash S$
    in lb, replace B by the smaller blocks $S \cup C_i$, $1 \leq i \leq p$
  `end_while`
  `return` $H = G_{lb}$

Theorem 4 ([2], property 7.5). *The algorithm* **MCSep** *outputs a minimal triangulation H of G, and the maximal cliques of H are exactly the blocks of lb.*

Actually, if instead of choosing a minimum cardinality separator S of $R(B)$ we just take an arbitrary minimal separator of $R(B)$, we obtain an implementation of algorithm **MinimalTriangulation** of section 3. A chordal graph with n vertices has at most n minimal separators. Therefore, the algorithm performs at most n splitting operations, where n is the number of vertices of G.

The complexity of the algorithm is clearly polynomial. We do not go into further details, since for the graphs of bounded asteroidal number we will propose a slightly different and more efficient algorithm.

6 Graphs with bounded asteroidal number

An *asteroidal set* of a graph G is a set of pairwise non adjacent vertices A such that for each $x \in A$, there is one connected component of $G\backslash N(x)$ containing all the vertices of $A\backslash\{x\}$. The asteroidal number $\mathrm{an}(G)$ of the graph G is the maximal cardinality of an asteroidal set of G.

We show in this section that, for a class of graphs with bounded asteroidal number, the minimum cardinality separator strategy gives a constant factor approximation for treewidth. More precisely, for any graph G, the algorithm **MCSep** produces a triangulation such that $\mathrm{tw}(H) \leq 8\,\mathrm{an}(G) \times \mathrm{tw}(G)$.

In any graph, the cardinality of a blocking set is bounded by the asteroidal number [7].

Proposition 3. *For any block B of G, $|\mathcal{BS}(B)| \leq \mathrm{an}(G)$.*

In a graph of small asteroidal number, each blocking set is small. We use this fact to show that, if the minimal separators in the blocking set of B are "small" and the block is "large", then the block can be splitted using a "small" minimal separator. Thus, the **MCSep** heuristic will keep choosing small minimal separators (of size at most $4\,\mathrm{tw}(G)$) until all the blocks become of size at most $8\,\mathrm{an}(G)\,\mathrm{tw}(G)$. We conclude that our strategy gives a good approximation for the treewidth of graphs with small asteroidal number.

Definition 5. *Let B be a block of G. We denote*

$$\mathrm{Border}(B) = \{x \in B \mid x \textit{ has a neighbor in } G \backslash B\} \textit{ and } \mathrm{Int}(B) = B \backslash \mathrm{Border}(B).$$

So $\mathrm{Border}(G) = \emptyset$, $\mathrm{Border}((S,C)) = S$ and $\mathrm{Border}(P(S_1,\ldots,S_p)) = S_1 \cup \cdots \cup S_p$.

Proposition 4. *Let G be a graph of treewidth at most k. Let B be a block of G with $|B| \geq 8k$. If $|\mathrm{Int}(B)| \geq |B|/2$, there is a minimal separator $S \subseteq B$ of $R(B)$ such that $|S| \leq 4k$.*

Proof. We show first that there is a vertex $x \in \mathrm{Int}(B)$ such that $|N_G(x)| \leq 4k$. Let $m(B)$ be the number of edges of $G[B]$. We count $m(B)$ in two different ways. Since $G[B]$ is an induced subgraph of G, its treewidth is at most k. It is well-known that in a graph of treewidth at most k, the number of edges is at most k times the number of vertices. Therefore, $m(B) \leq k|B|$. Suppose now that each vertex x of $\mathrm{Int}(B)$ has strictly more than $4k$ neighbors in G, we show that $m(B) > k|B|$. Since $x \in \mathrm{Int}(B)$, we have $N_G(x) \subseteq B$, thus the number of edges of $G[B]$ incident to at least one vertex of $\mathrm{Int}(B)$ is strictly greater than $4k|\mathrm{Int}(B)|/2 = 2k|\mathrm{Int}(B)|$. But $|\mathrm{Int}(B)| \geq |B|/2$, so $m(B) > 2k|\mathrm{Int}(B)| \geq k|B|$. This contradicts $m(B) \leq k|B|$.

We have proved that there is an $x \in \mathrm{Int}(B)$ such that $N_G(x) \leq 4k$. Let y be a vertex of $B \backslash N_G(x)$, different from x (y exists because $|B| > 8k$). Clearly $N_G(x)$ separates x and y in G, so there is a minimal separator $S \subseteq N_G(x)$ separating x and y in G. Then $|S| \leq 4k$ and, by theorem 3, S is a minimal separator of the realization $R(B)$ of B. □

Corollary 1. *Let G be a graph and let $k = \mathrm{tw}(G)$. Consider a block B of G such that all the minimal separators bordering B are of size at most $4k$. If the blocking set of B has at most a elements and $|B| > 8ka$, there is a minimal separator S of $R(B)$ such that $|S| \leq 4k$.*

Proof. Let $\mathcal{BS}(B) = \{S_1,\ldots,S_p\}$ be the blocking set of B, by proposition 3 $\mathcal{BS}(B)$ has at most $\mathrm{an}(G)$ elements. We have $\mathrm{Border}(B) = S_1 \cup \cdots \cup S_p$, and each S_i has at most $4k$ vertices, so $|\mathrm{Border}(B)| \leq 4ka$. Thus, $|\mathrm{Int}(B)| = |B| - |\mathrm{Border}(B)| > |B|/2$. The existence of the minimal separator S of $R(B)$ of size at most $4k$ separating follows directly from proposition 4. □

Theorem 5. *Let G be a graph of treewidth k and of asteroidal number at most a. The* MCSep *strategy gives a triangulation of G of width at most $8ak$.*

Proof. Suppose that, at the end of the algorithm, there is a bock B in the list lb such that $|B| > 8ak$. Notice that at least one of the minimal separators bordering B is of size greater than $4k$. Indeed, if all the minimal separators bordering B are of size at most $4k$, according to proposition 4, $R(B)$ has a minimal separator of size at most $4k$. This contradicts the fact that, at the end of the algorithm, the realization of each block in lb is a clique.

Consider the decreasing sequence of blocks $B_0 = G \supset B_1 \supset \cdots \supset B_p = B$ containing B during the execution of **MCSep**. Let B_i be the first block of the sequence such that one of the minimal separators bordering B_i is of size greater than $4k$. Clearly, $1 \leq i \leq p$. The block B_{i-1} was split by using a minimum cardinality separator S of $R(B_{i-1})$. We show that $|S| > 4k$. The block B_i is of form $S \cup C$, where C is a the connected component of $R(B_{i-1}) \backslash S$. Let T denote a minimal separator bordering B_i such that $|T| > 4k$ (T exists by choice of B_i). According to theorem 3, T is one of the minimal separators bordering B_{i-1} or $T \subseteq S$. By choice of B_i, all the minimal separators bordering B_{i-1} are of size at most $4k$, so the only possibility is $T \subseteq S$. Thus, $|S| > 4k$.

Since $B \subset B_{i-1}$ and $|B| > 8ak$ we have $|B_{i-1}| > 8ak$. Since all the minimal separators bordering B are of size at most $4k$, by corollary 1 we know there is a minimal separator S' of $R(B_{i-1})$ of size at most $4k$. But $|S| > 4k$, contradicting the fact that S is a minimum cardinality separator of $R(B_{i-1})$. □

Computing the asteroidal number of an arbitrary graph is NP-hard [11], which seems to be a major inconvenient for our algorithm. We can transform the algorithm such that, given a graph G and a number a it correctly outputs one of the following:

1. a triangulation of width at most $8a \times \mathrm{tw}(G)$.
2. G has asteroidal number strictly greater than a.

The algorithm should work the following way: we use the `MCSep` strategy and, for each block B obtained during the algorithm, we count the size of its blocking set $\mathcal{BS}(B)$. If $|\mathcal{BS}(B)| > a$, we output $\mathrm{an}(G) > a$. Otherwise, according to corollary 1, $|B| \leq 8a\,\mathrm{tw}(G)$ or $R(B)$ has a minimal separator of size at most $4\,\mathrm{tw}(G)$. Consequently, the triangulation produced by the algorithm is of width at most $8a\,\mathrm{tw}(G)$.

Now let us discuss a simpler and more efficient algorithm for approximating the treewidth of graphs with bounded asteroidal number. The `MCSep` strategy chooses, for each block B of the list of blocks, a separator S of $R(B)$ of minimum size. For our purpose, it is sufficient to find a minimal separator S of $R(B)$ such that $|S| \leq 4\,\mathrm{tw}(G)$. Observe that, in proposition 4, we have not only proved that there is a minimal separator S of $R(B)$ such that $|S| \leq 4k$, but we have also shown there is a vertex $x \in \mathrm{Int}(B)$ such that $|N_{R(B)}(x)| \leq 4k$. Thus, for finding a separator S of $R(B)$ of size at most $4k$, it is sufficient to choose the vertex x of $R(B)$ minimizing $|N_{R(B)}(x)|$ and to take a minimal separator $S \subseteq N_{R(B)}(x)$. We use the following lemma :

Lemma 1 ([9]). *Let $G = (V, E)$ be a graph and x be a vertex of G. For each connected component C of $G \backslash (N_G(x) \cup \{x\})$, its neighborhood $N_G(C)$ is a minimal separator of G.*

Applying this lemma to the graph $R(B)$, one can find a minimal separator S of $R(B)$ contained in $N_{R(B)}(x)$ in time $\mathcal{O}(n + m(R(B)))$, where n is the number of vertices of G and $m(R(B))$ is the number of edges of $R(B)$. Clearly, $m(R(B)) \leq n^2$. Once we found the minimal separator S, we can split the block B into smaller blocks in time $\mathcal{O}(n + m(R(B)))$, so $\mathcal{O}(n^2)$.

The algorithm uses at most n splitting operations, so the global cost of the algorithm is $\mathcal{O}(n^3)$. We have proved :

Theorem 6. *Given a graph $G = (V, E)$ and a number a, there is an algorithm which correctly outputs that $\mathrm{an}(G) > a$, or a triangulation of G of width at most $8a\,\mathrm{tw}(G)$. The complexity of the algorithm is $\mathcal{O}(n^3)$, where $n = |V|$.*

7 `MCSep` is not a constant factor approximation algorithm for the treewidth problem

A natural question is to ask whether the `MCSep` strategy is a constant factor approximation for the treewidth of arbitrary graphs. Unfortunately this is not the case. We prove that, for any constant c, there is a graph G on which the *MCSep* strategy yields a tree-decomposition whose width is more than $c\,\mathrm{tw}(G)$.

Consider a clique on q vertices $I_1, I_2, \ldots, I_q$. Divide each edge of the clique by a vertex M_{ij}. Add a vertex C adjacent to $I_1, \ldots, I_q$. Finally, replace C by a clique of size $2p$ and each vertex I_i by an independent set of size p. Take p "much bigger" than q. This graph is denoted by $G_{p,q}$, or simply G (see figure 1 for $q = 3$).

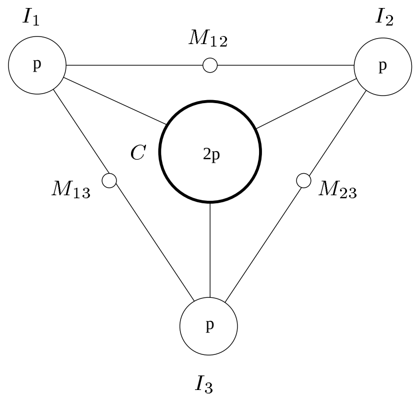

Fig. 1. $G_{p,q}$

Proposition 5. *The minimum cardinality separator strategy gives a triangulation $H_{p,q}$ of $G_{p,q}$ such that* $\mathrm{tw}(H_{p,q}) \geq (q+2)p - 1$.

Proof. The minimum size separators are exactly the separators of type $I_i \cup I_j$, which have size $2p$. Moreover, they are pairwise parallel. By proposition 1 our strategy choses to complete all these minimal separators, obtaining a clique $\Omega = C \cup I_1 \cup I_2 \cup \cdots \cup I_q$, of size $(q+2)p$. □

Proposition 6. *The treewidth of $G_{p,q}$ is at most $2p + q(q-1)/2$.*

Proof. Let $\Omega = C \cup \{M_{ij} \mid 1 \leq i < j \leq q\}$. Then the graph $H = G_\Omega$ obtained from G by completing Ω into a clique is a triangulation of G. Indeed, Ω separates in H any two vertices of $V \backslash \Omega$, so it is the only minimal separator of H. The cliquesize of H graph is $|\Omega| + 1$, so $\mathrm{tw}(G) \leq |\Omega| = 2p + q(q-1)/2$. □

For any p and q such that $p > q^2$, we have $\mathrm{tw}(H_{p,q}) > \frac{q}{2}\,\mathrm{tw}(G_{p,q})$. We conclude that the minimum cardinality separator strategy is not a constant factor approximation for the treewidth problem.

8 A $\mathcal{O}(k \log k)$ approximation algorithm for treewidth

We present in this section a polynomial algorithm that, given any graph G, outputs a tree-decomposition of G of width $\mathcal{O}(k \log k)$, where k is the treewidth of G.

In [3], Bodlaender et al. give an $\mathcal{O}(\log n)$ approximation algorithm for the treewidth of arbitrary graphs. Their algorithm uses the notion of ρ-separator.

Definition 6. *Let G be a graph and W a set of vertices of G. Consider a number ρ, $0 < \rho < 1$. A ρ* separator *of W in G is a set of vertices S such that each connected component of $G \backslash S$ contains at most $\rho|W|$ vertices of W.*

Theorem 7 ([3]). *Let G be a graph of treewidth at most k. For any set of vertices W, there is a $\frac{1}{2}$-separator of W of size at most $k+1$.*

If, for any graph G and any set of vertices W, we could compute a $\frac{1}{2}$-separator of W of minimum size in polynomial time, this would lead to a 3-approximation algorithm for the treewidth of arbitrary graphs. There exists several approximation results for computing a ρ-separator of small size, one of them was used in [3] in order to obtain the $\mathcal{O}(\log n)$ approximation algorithm for the treewidth of graphs. A recent result of Even et al. [8] allows us to improve the $\mathcal{O}(\log n)$ factor to $\mathcal{O}(\log k)$, where k is the treewidth of the input graph.

Theorem 8. *Let G be a graph and W a set of vertices of G. Let τ be the size of the smallest ρ separator of W in G, $1/2 \leq \rho < 1$. Given ρ', $\rho < \rho' \leq 1$, there is a polynomial algorithm computing a ρ'-separator of size at most $c_{\rho,\rho'}\tau \log \tau$, where $c_{\rho,\rho'}$ is a constant depending only on ρ and ρ'.*

In the paper of Even et al. the theorem is stated in terms of ρ-edge separators, but, as they point out, a vertex separator in an undirected graph can be viewed as an edge separator in an appropiate directed graph, so we refer directly to their result concerning edge separators in directed graphs. The size of the separator computed by their algorithm is at most $\frac{\rho'(1+1/\delta)}{\rho'-\rho}\ln(\frac{\rho'(1+\delta)}{\rho'-\rho}\tau)\tau$, where δ is an arbitrary positive value.

We use the algorithm of [3] and theorem 8 for computing a tree-decomposition of G of width $\mathcal{O}(k \log k)$, where $k = \mathrm{tw}(G)$.

The tree-decomposition of $G = (V, E)$ is obtained by calling `Makedec`$(G, \emptyset, V)$.

procedure `Makedec`(G, S, C)
`Input:` G and two disjoint sets of vertices S and C.
`Output:` a tree decomposition of $G_S[S \cup C]$.

`if` $|C| \leq \rho'|S|$ `then`
  `return` a tree-decomposition with a single node, labeled $S \cup C$.
`else`
  find a ρ'-separator S' of S in G with the algorithm of theorem 8
  `if` $S' = S$ `then`
    let x be a vertex of C
    $S_1 = S \cup \{x\}$; $C_1 = C\backslash\{x\}$
    `return Makedec`(G, S_1, C_1)
  `end if`
  Compute the connected components $C_1, \ldots C_q$ of $G[C\backslash S']$
  `for` $p = 1$ `to` q `do`
    $S_p \leftarrow N_G(C_p)$
    `call Makedec`(G, S_p, C_p)
  `end for`
  `return` a tree-decomposition having a root r labelled $S \cup S'$,
    whose children are the tree-decompositions returned by the calls
    of `Makedec`(G, S_p, C_p)
`end if`

Let $c_{\rho'} = c_{\frac{1}{2},\rho'}$, the constant involved in the algorithm of theorem 8 for $\rho = \frac{1}{2}$.

Proposition 7. *For any graph $G = (V, E)$,* `Makedec`(G, S, C) *returns a tree-decomposition of $G[S \cup C]$, such that the root node contains all the vertices of S.*

If $|S| \leq \frac{c_{\rho'}}{1-\rho'} k \log k$, where $k = \mathrm{tw}(G) + 1$, then the width of the decomposition is at most $\frac{1+\rho'}{1-\rho'} c_{\rho'} k \log k$.

Proof. Our algorithm is almost identical to the one of [3], so the proof of our proposition is very similar to the proof of [3], Claim 5.1.

We prove our statement by induction on the recursive structure of the `Makedec` procedure. The proposition is clearly true if $|C| \leq \rho'|S|$. Clearly, each vertex $x \in S \cup C$ will appear in at least one label of the returned labelled tree. Let us show that for each edge xy of $G[S \cup C]$, there is a node of the returned labelled tree whose label contains both x and y. If $x, y \in S \cup S'$ then x and y will be in the label of the root node. Otherwise, there is a connected component C_p of $G[C\backslash S']$ containing x

or y, so the edge xy is in $G[S_p \cup C_p]$. By induction hypothesis, there is a node j in the tree-decomposition returned by $\texttt{Makedec}(G, S_p, C_p)$ such that $x, y \in X_j$.

We denote $(T = (I, F), \mathcal{X})$ the labelled tree returned by our algorithm. We now prove that, for any vertex $x \in S \cup C$, the set of nodes $\{i \in I \mid x \in X_i\}$ forms a connected subtree of T. If $x \notin S \cup S'$, then x is in some component C_p of $G[C \backslash S']$ and the nodes of T containing x are exactly the nodes of $\texttt{Makedec}(G, S_p, C_p)$ containing x, so the result holds by induction. Otherwise, for each component C_p of $G[C \backslash S']$, the nodes containing x in the tree-decomposition $(T_p, \mathcal{X}_p)$ returned by $\texttt{Makedec}(G, S_p, C_p)$ form a subtree of T_p. If this subtree is not empty, since $x \notin C_p$, we have that $x \in S_p$ so x appears in the label of the root node of T_p. Since the root of T_p is a son of the root of T, we conclude that $\{i \in I \mid x \in X_i\}$ forms a connected subtree of T.

Finally, we show that the labels of the returned tree-decomposition are of size at most $\frac{1+\rho'}{1-\rho'} c_{\rho'} k \log k$. By induction, it is sufficient to show that the label of the root is of size at most $\frac{1+\rho'}{1-\rho'} c_{\rho'} k \log k$ and that each set S_p is of size at most $\frac{c_{\rho'}}{1-\rho'} k \log k$.

According to theorem 7, for any set of vertices S of G there is a $\frac{1}{2}$-separator S_{opt} of S in G such that $|S_{\text{opt}}| \leq k$. By theorem 8, the ρ'-separator S' of S is of size at most $c_{\rho'} k \log k$. Consider the case when $S' = S$. Then $|S_1| = |S| + 1$. The constant $c_{\rho'}$ is greater than one (see [8]) and $1/2 < \rho' < 1$, so $\frac{c_{\rho'}}{1-\rho'} k \log k \geq 2c_{\rho'} k \log k \geq c_{\rho'} k \log k + 1$, and therefore $|S_1| \leq \frac{c_{\rho'}}{1-\rho'} k \log k$. Suppose we are in the case $S' \neq S$. For each connected component C_p of $G[C \backslash S']$, the neighborhood of C_p in G is contained in $S \cup S'$. Let $\tilde{S}_p = S \cap N_G(C_p)$. Then all the vertices of $\tilde{S}_p$ are in a same connected component of $G \backslash S'$, and since S' is a ρ'-separator of S we have that $|\tilde{S}_p| \leq \rho' |S|$. Therefore, $|S_p| \leq |\tilde{S}_p| + |S'| \leq (\frac{\rho'}{1-\rho'} + 1) c_{\rho'} k \log k$, so $|S_p| \leq \frac{c_{\rho'}}{1-\rho'} k \log k$. The label of the root node is $S \cup S'$, so its size is at most $\frac{1+\rho'}{1-\rho'} c_{\rho'} k \log k$. $\square$

Thus, we have obtained:

Theorem 9. *There is a polynomial algorithm that, given a graph G, computes a tree-decomposition of G of width $\mathcal{O}(k \log k)$, where $k = \text{tw}(G) + 1$.*

References

1. R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network flows: theory, algorithms, and applications.* Prentice Hall, Englewood Cliffs NJ, 1993.
2. A. Berry. *Désarticulation d'un graphe.* PhD thesis, Université Montpellier II, 1998.
3. H. Bodlaender, J.R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating treewidth, pathwidth, and minimum elimination tree height. *J. of Algorithms*, 18:238–255, 1995.
4. V. Bouchitté and I. Todinca. Approximating the treewidth of AT-free graphs. In *Proceedings 26th Workshop on Graph-Theoretic Aspects in Computer Science (WG 2000)*, volume 1928 of *Lecture Notes in Computer Sciences*, pages 59–70. Springer-Verlag, 2000.
5. V. Bouchitté and I. Todinca. Listing all potential maximal cliques of a graph. In *Proceedings 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2000)*, volume 1770 of *Lecture Notes in Computer Science*, pages 503–515. Springer-Verlag, 2000.
6. V. Bouchitté and I. Todinca. Treewidth and minimum fill-in: grouping the minimal separators. *SIAM J. on Computing*, 31(1):212 – 232, 2001.
7. H. Broersma, T. Kloks, D. Kratsch, and H. Müller. A generalization of AT-free graphs and a generic algorithm for solving triangulation problems. In *Workshop on Graph-theoretic Concepts in Computer Science WG'98*, volume 1517 of *Lecture Notes in Computer Science*, pages 88–99. Springer-Verlag, 1998.

8. G. Even, J. Naor, S. Rao, and B. Schieber. Fast approximate graph partitioning algorithms. *SIAM J. on Computing*, 28(6):2187–2214, 1999.
9. T. Kloks and D. Kratsch. Listing all minimal separators of a graph. *SIAM J. Comput.*, 27(3):605–613, 1998.
10. T. Kloks, D. Kratsch, and H. Müller. Approximating the bandwidth for asteroidal triple-free graphs. *Journal of Algorithms*, 32:41–57, 1999.
11. T. Kloks, D. Kratsch, and H. Müller. On the structure of graphs with bounded asteroidal number. *Graphs and Combinatorics*, 17:295–306, 2001.
12. A. Parra and P. Scheffler. Characterizations and algorithmic applications of chordal graph embeddings. *Discrete Appl. Math.*, 79(1-3):171–188, 1997.