Discrete Applied Mathematics 157 (2009) 2718–2725

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Branchwidth of chordal graphs

Christophe Paul [a,*], Jan Arne Telle [b,1]

[a] CNRS, LIRMM, 161 rue Ada, 34392 Montpellier Cedex 2, France

[b] Department of Informatics, University of Bergen, Norway

ARTICLE INFO

Article history:
Received 19 December 2005
Received in revised form 1 October 2007
Accepted 6 August 2008
Available online 20 February 2009

Keywords:
Graph decomposition
Width parameter
Algorithms
Graphs classes

ABSTRACT

This paper revisits the 'branchwidth territories' of Kloks, Kratochvíl and Müller [T. Kloks, J. Kratochvíl, H. Müller, New branchwidth territories, in: 16th Ann. Symp. on Theoretical Aspect of Computer Science, STACS, in: Lecture Notes in Computer Science, vol. 1563, 1999, pp. 173–183] to provide a simpler proof, and a faster algorithm for computing the branchwidth of an interval graph. We also generalize the algorithm to the class of chordal graphs, albeit at the expense of exponential running time. Compliance with the ternary constraint of the branchwidth definition is facilitated by a simple new tool called k-troikas: three sets of size at most k each are a k-troika of set S, if any two have union S. We give a straightforward $O(m + n + q^2)$ algorithm, computing branchwidth for an interval graph on m edges, n vertices and q maximal cliques. We also prove a conjecture of Mazoit [F. Mazoit, A general scheme for deciding the branchwidth, Technical Report RR2004-34, LIP — École Normale Supérieure de Lyon, 2004. http://www.ens-lyon.fr/LIP/Pub/Rapports/RR/RR2004/RR2004-34.pdf], by showing that branchwidth can be computed in polynomial time for a chordal graph given with a clique tree having a polynomial number of subtrees.

1. Introduction

Branchwidth and treewidth are connectivity parameters of graphs, and whenever one of these parameters is bounded by some fixed constant on a class of graphs, then so is the other [16]. Since many graph problems that are in general NP-hard can be solved in linear time on such classes of graphs, both treewidth and branchwidth have played a large role in many investigations in algorithmic graph theory. Recently there has been a focus on branchwidth [7,5,4,8,9] to give e.g. good heuristics for the traveling salesman problem and fast parameterized algorithms for various types of optimization problems. These algorithms always involve a stage that constructs a branch-decomposition with small branchwidth, and another stage solving the problem using the decomposition by a running time depending heavily on its branchwidth. Efficient algorithms computing optimal branch-decompositions, as we give in this paper, could therefore be the crucial factor that can make or break the application.

The understanding of branchwidth of special graph classes is relatively limited. We give a brief overview of the literature. In a paper from 1994 Seymour and Thomas showed that branchwidth is NP-complete in general, and followed this by their celebrated ratcatcher method, computing branchwidth of planar graphs in polynomial time [17]. In 1997 Bodlaender and Thilikos used fairly brute-force methods to give a linear-time algorithm, deciding if a graph has branchwidth at most some constant k [1] and a very elegant algorithm for graphs of branchwidth 3 [2]. Then in 1999 Kloks, Kratochvíl and Müller [11,12] pushed into new territory, by showing that branchwidth is already NP-complete for split graphs (which is a subclass of chordal graphs) and bipartite graphs, with the bulk of their paper being an $O(n^3 \log n)$ algorithm for branchwidth of interval

* Corresponding author.
E-mail addresses: paul@lirmm.fr (C. Paul), telle@ii.uib.no (J.A. Telle).

[1] Research conducted while on sabbatical at LIRMM.

0166-218X/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2008.08.006

graphs with the comment that:

> *"it is somewhat surprising that this algorithm is by no means straightforward and its correctness requires a nontrivial proof".*

In contrast, we give a straightforward $O(m+n+q^2)$ algorithm, whose correctness proof is easy to follow, for branchwidth of an interval graph on m edges, n vertices and q maximal cliques. The basic idea of our algorithm is the same as the one in [11,12]. However, our algorithm was developed independently, using the concept of k-troikas that dramatically facilitate compliance with the ternary constraint in the definition of branchwidth: three sets of size at most k each are a k-troika of set S, if any two have union S. Recently, Mazoit gave a polynomial-time algorithm for branchwidth of circular-arc graphs, and conjectured that branchwidth can be computed in polynomial-time for chordal graphs given with a clique tree having a polynomial number of subtrees [13]. We prove his conjecture in this paper. Indeed, it follows by a generalization of the interval graph algorithm, since we show that branchwidth of a chordal graph with clique tree T can be found by simple dynamic programming over chordal supergraphs having a clique tree resulting from contracting edges of T. This algorithm will compute the branchwidth of any chordal graph, and it will do this in polynomial-time whenever T has a polynomial number of subtrees.

In Section 2 we give some standard definitions and some preliminary results from [15]. Section 3 is dedicated to the study of the central concept of k-troikas in a purely set-theoretic setting. In Section 4 we present a simple algorithm computing branchwidth for interval graphs, and more generally for chordal graphs with a clique tree having a polynomial number of subtrees.

2. Standard definitions and earlier results

We consider simple undirected and connected graphs G with vertex set $V(G)$, and edge set $E(G)$. We denote G subgraph of H by $G \subseteq H$ which means that $V(G) = V(H)$ and $E(G) \subseteq E(H)$, and we also say that H is a supergraph of G. For a set $A \subseteq V(G)$, $G(A)$ denotes the subgraph of G induced by the vertices in A. A is called a *clique* if $G(A)$ is complete. The set of neighbors of a vertex v in G is $N(v) = \{u \mid uv \in E(G)\}$. A vertex set $S \subset V(G)$ is a *separator* if $G(V(G) \setminus S)$ is disconnected. Given two vertices u and v, S is a u, v*-separator* if u and v belong to different connected components of $G(V(G) \setminus S)$. A u, v-separator S is *minimal* if no proper subset of S separates u and v. In general, S is a *minimal separator* of G if there exist two vertices u and v in G such that S is a minimal u, v-separator. A graph is *chordal* if it contains no induced cycle of length ≥ 4. A *triangulation* of a graph G is a chordal supergraph of G. In a *clique tree* of a chordal graph G the nodes are in 1-1 correspondence with the maximal cliques of G and the set of nodes whose maximal cliques contain a given vertex form a subtree. For further terminology, see e.g. [10]. We usually refer to nodes of a tree and vertices of a graph.

A *branch-decomposition* (T, μ) of a graph G is a tree T with nodes of degree one and three only, together with a bijection μ from the edge-set of G to the set of degree-one nodes (leaves) of T. For an edge e of T let T_1 and T_2 be the two subtrees resulting from $T \setminus \{e\}$, let G_1 and G_2 be the graphs induced by the edges of G, mapped by μ to leaves of T_1 and T_2 respectively, and let $mid(e) = V(G_1) \cap V(G_2)$. The width of (T, μ) is the size of the largest $mid(e)$ thus defined. For a graph G its *branchwidth* $bw(G)$ is the smallest width of any branch-decomposition of G.[2]

It has already been noted in different contexts (e.g. [12,14,15]) that for any graph G, there exists a chordal supergraph H of G, such that $bw(H) = bw(G)$. But this property is still far from a characterization of the branchwidth of a graph G in terms of triangulations of G, in particular it is vacuous in case G is a chordal graph. Very recently, Mazoit [14] and Paul and Telle [15] independently discovered two different such characterizations.

Theorem 1 (*[14]*). *For any graph G, let $\mathcal{H}$ be the set of its triangulations. Then*

$$bw(G) = \min_{H \in \mathcal{H}} \max\{bbw(X) \mid X \text{ maximal clique of } H\}.$$

Actually, Mazoit showed that it is enough to restriction attention to a subset of triangulations that he called *efficient triangulations*. The parameter $bbw(X)$ can be understood as a local branchwidth for the maximal clique X under the constraints of the graph G, but we refer to [14] for more details. This characterization enabled Fomin, Mazoit and Todinca to design an exact algorithm for computing branchwidth of a graph in time $O((2+\sqrt{3})^n n^{O(1)})$ [6].

Let us present the characterization of [15], which will be the basis of our algorithms. We first define k-troikas[3] and k-good chordal graphs, which are central tools in our investigation of branchwidth. The use of k-troikas for branchwidth allows the separation of purely set theoretic constraints from graph theoretic ones.

Definition 1 (*[15]*). A k-troika (A, B, C) of a set X are 3 subsets of X, called the three parts, such that $|A| \leq k$, $|B| \leq k$, $|C| \leq k$, and $A \cup B = A \cup C = C \cup B = X$. (A, B, C) *respects* $S_1, S_2, \ldots, S_q$ if any S_i, $1 \leq i \leq q$ is contained in at least one of A, B or C.

[2] The graphs of branchwidth 1 are the stars, and constitute a somewhat pathological case. To simplify we therefore restrict attention to graphs having branchwidth $k \geq 2$, in other words our statements are correct only for *graphs having at least two vertices of degree more than one.*

[3] A troika is a horse-cart drawn by three horses, and when the need arises, any two of them should also be able to pull the cart.

Definition 2 ([15]). A *k-good chordal graph* is a chordal graph in which every maximal clique X has a k-troika respecting the minimal separators contained in X.

Theorem 2 ([15]). *A graph G has branchwidth at most $k \Leftrightarrow G$ is subgraph of a k-good chordal graph.*

3. *k*-troikas

This section will be devoted to a study of the conditions, under which a set X has a k-troika respecting a given set of subsets. As with branchwidth, we restrict attention to the case $k \geq 2$. These conditions on the given sets, which will turn out to be testable by simple algorithms, will in conjunction with Theorem 2 be useful for designing algorithms computing branchwidth of graphs.

Observation 1. *If X has a k-troika respecting $S_1, S_2, \ldots, S_q$ then $|S_i| \leq k$ for each $1 \leq i \leq q$ and $|X| \leq \lfloor 3k/2 \rfloor$.*

The above is obvious, every subset must be of size at most k, since it must be contained in a part of size at most k, and the fact that every pair of parts must have union X, means that every element of X must belong to at least two parts which implies $2|X| \leq 3k$.

Note that the case of respecting a single subset is trivial, the necessary and sufficient conditions are that the subset has at most k elements, and $|X| \leq \lfloor 3k/2 \rfloor$. Likewise, if $|S_1 \cup S_2 \cup \cdots \cup S_q| \leq k$ then G has a k-troika respecting $S_1, S_2, \ldots, S_q$ precisely when $|X| \leq \lfloor 3k/2 \rfloor$ since we may as well view the union of all the subsets as a single subset. Finally, an observation that follows directly from the definition.

Observation 2. *If (A, B, C) is a k-troika of X respecting $S_1, \ldots, S_q$, then for any $X' \subseteq X$ and $S'_i \subseteq (S_i \cap X')$, $1 \leq i \leq q$ the triple $(A \cap X', B \cap X', C \cap X')$ is a k-troika of X' respecting $S'_1, \ldots, S'_q$.*

3.1. *k-Troikas respecting two subsets*

In this section, we consider conditions under which a set X has a k-troika respecting two subsets S_1, S_2. As mentioned above we assume that $|S_1 \cup S_2| > k$ and also wlog that any k-troika (A, B, C) respecting S_1, S_2 has $S_1 \subseteq A$ and $S_2 \subseteq B$. Note that if X has a k-troika respecting S_1, S_2, then it has one where no element of X belongs to all three parts. The constraints mentioned above motivates the following definition.

Definition 3. A k-tripartition of a set X is a partition of X into three (disjoint) partition classes, such that the sum of sizes of any two partition classes is at most k. A k-tripartition (T_1, T_2, T_3) of X *respects* S_1, S_2 if $S_1 \subseteq T_1 \cup T_3$, $S_2 \subseteq T_2 \cup T_3$, and $S_1 \cap S_2 \subseteq T_3$.

Observation 3. *If (T_1, T_2, T_3) is a k-tripartition of X then $(T_1 \cup T_3, T_2 \cup T_3, T_2 \cup T_1)$ is a k-troika of X, and the former respects S_1, S_2 iff the latter does. Conversely, if (A, B, C) is a k-troika of X with $A \cap B \cap C = \emptyset$ then $(A \cap C, B \cap C, B \cap A)$ is a k-tripartition of X, and the former respects S_1, S_2 (with $S_1 \subseteq A$, $S_2 \subseteq B$ and $|S_1 \cup S_2| > k$ as discussed above) iff the latter does.*

In view of this observation, when it comes to k-troikas respecting two subsets S_1, S_2, we need only consider those that arise from k-tripartitions where one of the partition classes contains the intersection of the two subsets. In Observation 1 we gave some obviously necessary conditions on $|X|, |S_1|, |S_2|$. What other necessary conditions do we have? Let us consider the case $|X| = 3k/2$ and k even. In this case only a 'balanced' k-tripartition with each partition class having $k/2$ vertices will do. Since we require $S_1 \cap S_2 \subseteq T_3$, the subcase where $|S_1 \cap S_2| > k/2$ therefore implies a stronger size restriction on X. The best we could hope for in this subcase is to set $T_3 = S_1 \cap S_2$ and put $k - |S_1 \cap S_2|$ vertices into each of T_1 and T_2 which yields the general statement:

Observation 4. *If X has a k-troika respecting S_1, S_2 then $|X| \leq |S_1 \cap S_2| + 2(k - |S_1 \cap S_2|) = 2k - |S_1 \cap S_2|$*

Note that we did not need to preface this observation by the condition "if $|S_1 \cap S_2| > k/2$" since $|X| \leq \lfloor 3k/2 \rfloor$ and $|S_1 \cap S_2| \leq k/2$ together imply $|X| \leq 2k - |S_1 \cap S_2|$. As the next theorem shows, these obviously necessary conditions are also sufficient (ONCAS).

Theorem 3. *A set X has a k-troika respecting S_1, S_2 (assume $|S_1 \cup S_2| > k$) if and only if $|X| \leq \lfloor 3k/2 \rfloor$, $|S_1| \leq k$, $|S_2| \leq k$ and $|X| \leq 2k - |S_1 \cap S_2|$.*

Proof. The necessity of these conditions have already been argued for. We prove that they are sufficient by considering two cases: $|S_1 \cap S_2| \leq k/2$ and $|S_1 \cap S_2| > k/2$. In the first case we can construct a 'balanced' k-tripartition (T_1, T_2, T_3) where each partition class has at most $k/2$ elements. For the vertices in $S_1 \cap S_2$ we put them all in T_3. For the vertices in $S_1 \setminus S_2$, we put up to $k/2$ of them in T_1 and the remainder in T_3. For the vertices in $S_2 \setminus S_1$, we put up to $k/2$ of them in T_2 and the remainder in T_3. The conditions $|X| \leq \lfloor 3k/2 \rfloor$, $|S_1| \leq k$, $|S_2| \leq k$, and $|S_1 \cap S_2| \leq k/2$ will ensure that each of T_1, T_2, T_3 constructed so

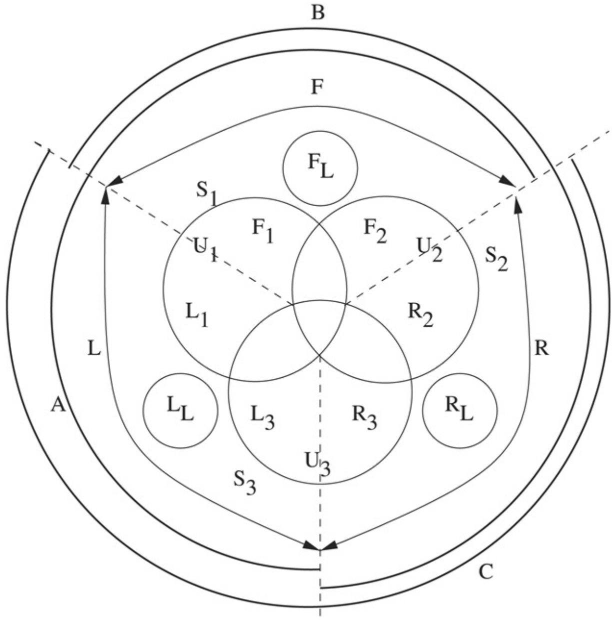

Fig. 1. Lemma 1. A 3-set system, with names as in the proof of Lemma 1.

far has at most $k/2$ elements. The vertices in $X \setminus S_1 \cup S_2$ are now put into T_1, T_2 or T_3 freely while simply ensuring that each partition class has at most $k/2$ elements, which is doable since $|X| \leq \lfloor 3k/2 \rfloor$ (note that if k is odd then '$\leq k/2$', 'up to $k/2$' and 'at most $k/2$' is the same as $\leq \lfloor k/2 \rfloor$.)

We turn to the case $|S_1 \cap S_2| > k/2$. Let $f_1 = k - (|S_1 \cap S_2| + |S_1 \setminus S_2|)$ and $f_2 = k - (|S_1 \cap S_2| + |S_2 \setminus S_1|)$. Note that $|X| - |S_1 \cup S_2| \leq 2k - |S_1 \cap S_2| - |S_1 \cup S_2| = f_1 + f_2$ where the first inequality comes from $|X| \leq 2k - |S_1 \cap S_2|$. Thus we can partition $X \setminus (S_1 \cup S_2)$ into F_1 and F_2 of sizes at most f_1 and at most f_2 respectively. The desired k-tripartition is then $T_3 = S_1 \cap S_2$, $T_1 = (S_1 \setminus S_2) \cup F_1$, $T_2 = (S_2 \setminus S_1) \cup F_2$. $\square$

Corollary 1. *The smallest k such that X has a k-troika respecting S_1, S_2 is*

$$\max \left\{ \begin{array}{l} |S_1|, |S_2|, \lceil 2|X|/3 \rceil, \\ \min\{|S_1 \cup S_2|, (\lceil |X| + |S_1 \cap S_2|)/2 \rceil\} \end{array} \right\}$$

and can be computed in constant time given $|S_1|, |S_2|, |X|, |S_1 \cap S_2|$.

Note that $|S_1 \cup S_2|$ is easily found from $|S_1|, |S_2|, |S_1 \cap S_2|$. The two terms inside the minimum covers the two cases where the resulting smallest k-troika (A, B, C) has either $S_1 \cup S_2 \subseteq A$ or $S_1 \subseteq A$ and $S_2 \subseteq B$, respectively. Let us remark that for the interval graph algorithm the above Corollary suffices, since we then only deal with 2 minimal separators for each maximal clique.

3.2. *k-troikas respecting q subsets*

We first consider the case of a set X respecting three subsets S_1, S_2, S_3, and denote by L the elements of X not belonging to any subset and by U_i, $1 \leq i \leq 3$ the elements belonging to S_i only: $L = X \setminus (S_1 \cup S_2 \cup S_3)$, $U_1 = S_1 \setminus (S_2 \cup S_3)$, $U_2 = S_2 \setminus (S_1 \cup S_3)$, $U_3 = S_3 \setminus (S_2 \cup S_1)$ (see Fig. 1).

Lemma 1. *X has a k-troika A, B, C with $S_1 \subseteq A, S_2 \subseteq B, S_3 \subseteq C$ $\Leftrightarrow$ the following system of linear equations in 5 non-negative integer variables a, b, c, d, e has a solution:*

$$\begin{array}{l} a \leq |U_1|; \qquad b \leq |U_2|; \qquad c \leq |U_3|; \qquad d + e \leq |L| \\ |S_3| + |U_2| + a - b + d + e \leq k \\ |S_1| + |U_3| + |L| + b - c - e \leq k \\ |S_2| + |U_1| + |L| - a + c - d \leq k. \end{array}$$

Proof. $\Leftarrow$: Partition U_1 into L_1, F_1 with $|L_1| = a$ and $|F_1| = |U_1| - a$. Partition U_2 into F_2, R_2 with $|F_2| = b$ and $|R_2| = |U_2| - b$. Partition U_3 into R_3, L_3 with $|R_3| = c$ and $|L_3| = |U_3| - c$. Partition L into F_L, R_L, L_L with $|F_L| = d$ and $|R_L| = e$ and $|L_L| = |L| - d - e$.

Then let $A = S_1 \cup L_3 \cup F_2 \cup F_L \cup L_L$, let $B = S_2 \cup R_3 \cup F_1 \cup F_L \cup R_L$, and let $C = S_3 \cup L_1 \cup R_2 \cup L_L \cup R_L$.

The system of equations guarantees that the cardinalities of A, B, C are at most k, and by construction we have $A \cup B = B \cup C = A \cup C = X$ and $S_1 \subseteq A, S_2 \subseteq B, S_3 \subseteq C$.

$\Rightarrow$: Note that if X has the desired k-troika then it has one with $A \cap B \cap C = S_1 \cap S_2 \cap S_3$. Let $L_1 = C \cap U_1$, let $F_1 = B \cap U_1$, let $F_2 = A \cap U_2$, let $R_2 = C \cap U_2$, let $R_3 = B \cap U_3$, and let $L_3 = A \cap U_3$. Furthermore, let $F_L = L \cap A \cap B$, let $L_L = L \cap A \cap C$, and let $R_L = L \cap B \cap C$.

It follows that $A = S_1 \cup L_3 \cup F_2 \cup F_L \cup L_L$, that $B = S_2 \cup R_3 \cup F_1 \cup F_L \cup R_L$, and $C = S_3 \cup L_1 \cup R_2 \cup L_L \cup R_L$.

Since the cardinalities of A, B, C are at most k, we must have $a = |L_1|, b = |F_2|, c = |R_3|, d = |F_L|, e = |R_L|$ a solution to the system of equations. $\square$

The only other possibility is that the union of two of the subsets is at most k, and in this case we may appeal to the conditions for respecting two subsets, giving:

Lemma 2. *X has a k-troika respecting S_1, S_2, S_3 $\Leftrightarrow$ it has one satisfying the conditions of Lemma 1 or it has one where either $S_1 \cup S_2, S_3$ or $S_1 \cup S_3, S_2$ or $S_2 \cup S_3, S_1$ satisfies the conditions of Theorem 3.*

To respect $q > 3$ subsets, we simply note that since each subset must be contained in one of the three parts of the k-troika, there must exist a partition of the subsets into three classes, such that every subset in the same class is contained in the same part.

Theorem 4. *X has a k-troika respecting $S_1, S_2, \ldots, S_q$ $\Leftrightarrow$ there exists a partition of $\{1, 2, \ldots, q\}$ into three classes P_1, P_2, P_3 such that by Lemma 2 X has a k-troika respecting the 3 subsets $W_1 = \bigcup_{i \in P_1} S_i$, $W_2 = \bigcup_{i \in P_2} S_i$, $W_3 = \bigcup_{i \in P_3} S_i$.*

Since a set of size q has 3^q partitions into three classes we have:

Corollary 2. *In time $O(poly(|X|)3^q)$ we can decide if a set X has a k-troika respecting subsets $S_1, S_2, \ldots, S_q$.*

4. Computing branchwidth of chordal graphs

Throughout this section, G is a chordal graph with m edges, n vertices, maximal cliques $\{X_1, X_2, \ldots, X_q\}$, having a clique-tree T_G with nodes $\{1, 2, \ldots, q\}$, such that node i corresponds to maximal clique X_i. When contracting an edge ij of clique-tree T_G we let the new tree node correspond to vertex set $X_i \cup X_j$. Let us first define the notion of *merged supergraph of a chordal graph* by way of edge contractions in a clique-tree.

Definition 4. A chordal graph H is a *merged supergraph* of a chordal graph G if H has a clique-tree T_H, that results from edge-contractions in some clique tree T_G of G.

To find the branchwidth k of G it suffices to search for k-good chordal graphs among the merged supergraphs of G. The rest of this subsection is devoted to a proof of this fact.

Lemma 3. *Let G be a chordal graph of $bw(G) = k$ which is not k-good, and let T_G be a clique-tree of G. Then any k-good chordal supergraph H of G has a maximal clique that contains two neighboring maximal cliques of T_G.*

Proof. As G is not k-good, it has a maximal clique X, for which there does not exist a k-troika respecting the minimal separators $S_1, S_2, \ldots, S_l$ contained in X. Let $X_1 \ldots X_l$ be the neighboring maximal cliques of X with $S_i = X \cap X_i$ the corresponding minimal separators ($1 \leqslant i \leqslant l$).

Let X' be a maximal clique of H containing X. Assume X_i, for some $1 \le i \le l$, is not contained in X'. Then there exists, for some $a \in X_i \setminus X'$ and $b \in X' \setminus X_i$, a minimal a, b-separator S_i' of H, such that $S_i \subseteq S_i' \subset X'$. Since H is a k-good chordal graph, the maximal clique X' has a k-troika respecting its minimal separators. If none of $X_1, X_2, \ldots, X_l$ were contained in X' then by Observation 2, X would have a k-troika respecting $S_1, S_2, \ldots, S_l$ contradicting the assumption. Therefore X' has to contain a neighboring maximal clique of X. $\square$

Let us make a few remarks about Lemma 3. Firstly, it holds for any clique-tree of G. Secondly, it follows from the proof that its statement can be strengthened as follows. For any maximal clique X' of H containing X, at least one minimal separator S_i of X has to be "killed": no minimal separator of X' contains S_i, and hence S_i is not contained in any minimal separator of H.

Lemma 4. *A chordal graph G has $bw(G) \leqslant k$ $\Leftrightarrow$ there exists a k-good chordal graph H that is a merged supergraph of G.*

Proof. $\Leftarrow$: By Theorem 2 the existence of a k-good chordal graph H that is a supergraph of G implies that $bw(G) \leqslant k$.

$\Rightarrow$: By induction on the number q of maximal cliques of G. By Theorem 2 G is the subgraph of a k-good chordal graph, and if $q = 1$ then G is a complete graph whose only supergraph G is a merged supergraph of it. For the inductive step, assume G is not a k-good chordal graph and let T_G be an arbitrary clique-tree of G. By Theorem 2 G has a k-good chordal supergraph H and by Lemma 3 there are two maximal cliques X_i, X_j in G that are neighbors in T_G with both contained in a single maximal clique of H. Let G' be the graph G with edges added to make $X_i \cup X_j$ into a clique. Note that $G' \subseteq H$ is a merged supergraph of G on fewer maximal cliques than G. By the induction hypothesis, there is a k-good chordal graph which is a merged supergraph of G' and therefore also of G. $\square$

Pre-processing (see below) to find $|S_i|, |X_i|, |S_i \cap S_j|, |X_{i,j}|$
For $1 \leq i \leq j \leq q+1$ **Do** Compute $K[i,j]$ by the formula of Corollary 1
$A[0] = 0$
For $j = 1$ to q **Do** $A[j] = min\{max\{A[i-1], K[i,j]\} : 0 < i \leqslant j\}$

Fig. 2. Computation of $bw(G) = A[q]$ for interval graph G.

The fact that a chordal graph G may not be $bw(G)$-good, constitutes a main difference between branchwidth and treewidth (the treewidth of a chordal graph is simply the size of its largest clique minus one). Actually, it has been proven that the branchwidth of split graphs, a subfamily of the chordal graphs having clique trees that are stars, is NP-complete [11,12].

4.1. Branchwidth of interval graphs

A graph is an interval graph iff it enjoys a *consecutive clique arrangement* (cca), *i.e.* an ordering of its maximal cliques $\mathcal{C}_{\mathcal{G}} = (X_1, \ldots, X_q)$ such that for any vertex x, the maximal cliques containing x occur consecutively. Notice that cca's are clique-trees that are paths. From any linear time interval graph recognition algorithm such a cca can be computed (see e.g. [3]). It is well known that for any $1 < i \leqslant q$, the set $S_i = X_{i-1} \cap X_i$ is a minimal separator of G. Let $S_1 = S_{q+1} = \emptyset$ be dummy separators. Let us denote by $X_{i,j} = \bigcup_{i \leqslant g \leqslant j} X_g$ $(1 \leqslant i \leqslant j \leqslant q)$ the union of vertex sets of consecutive cliques.

As Lemma 3 holds for arbitrary clique-trees, it holds also for cca's of interval graphs. After contracting an edge of a path we still have a path, so Lemma 4 can for interval graphs be rephrased as follows:

Corollary 3. *An interval graph G with cca $\mathcal{C}_{\mathcal{G}} = (X_1, \ldots, X_q)$ has $bw(G) \leqslant k \Leftrightarrow$ there exists a k-good interval graph H having cca $\mathcal{C}_H = (X'_1 \ldots X'_h)$ with $h \leq q$ such that for any $1 \leqslant i \leqslant h$, $X'_i = X_{l_i,r_i}$ with $l_1 = 1$, $l_i = r_{i-1} + 1$ for $i > 1$ and $r_h = q$.*

The cca $\mathcal{C}_H$ of the k-good merged supergraph H of G corresponds to what in [11,12] is called a *k-fragmentation* of $\mathcal{C}_G$ and the maximal cliques of H to what is there called k-fragments as they have a k-troika respecting their minimal separator (refer to [11,12] for definitions). It follows that Corollary 3 is a restatement of Theorem 25 of [12].

Let us now turn to the algorithmic part, and show how the complexity of the algorithm of [11,12] can easily be improved from $O(n^3 \log n)$ to $O(n^2)$ by a careful use of the structure of cca's.

Our algorithm first computes for each pair $1 \leq i \leq j \leq q$ the smallest value $K[i,j]$, such that if we merge the consecutive cliques $X_{i,j}$ into one big clique, it will have a $K[i,j]$-troika respecting S_i and S_{j+1}. This value is given by Corollary 1 (which can be compared with Definition 15 of [12].) Then by simple dynamic programming, we compute the best way of merging various such sets into a merged supergraph, see Fig. 2. Incrementally, in step j, we optimize over the possible cutoff points $1 \leq i \leq j$ that define the 'rightmost' merged set of cliques $X_{i,j}$. We prove correctness before considering the running time.

Theorem 5. *The computed value $A[q]$ is the branchwidth of interval graph G.*

Proof. Let us prove by induction that, for $1 \leqslant i \leqslant q$, $A[i] = bw(G_i)$ where G_i is the graph induced by $X_{1,i}$ with an extra dummy vertex x_i adjacent to S_{i+1}. By Corollary 1 $K[i,j]$ is the minimum such that set $X_{i,j}$ has a $K[i,j]$-troika respecting S_i and S_{j+1}. As $A[1] = K[1,1]$, X_1 has a $A[1]$-troika respecting S_2. Therefore $\{x_1\} \cup S_2$ also has a $A[1]$-troika respecting S_2. Theorem 2 implies that $bw(G_1) = A[1]$. Assume that $A[j-1] = bw(G_{j-1})$ for $j > 1$. Let H_j be the merged supergraph of G_j such that $bw(G_j) = bw(H_j)$. Then by Lemma 3 the maximal clique X_j is contained in H_j in a maximal clique $X' = X_{i,j}$ for some $1 \leqslant i \leqslant j$. It therefore follows from Corollary 3, that $bw(G_j) \leqslant \max\{A[i-1], K[i,j]\}$ for any $1 \leqslant i \leqslant j$, and thus $bw(G_j) = A[j]$. We proved that $bw(G_q) = A[q]$. Since G_q is the union of two connected components, the first one being G itself and the second an isolated vertex x_q, $bw(G) = bw(G_q)$. □

By Corollary 1 the computation of matrices K and A takes time $O(q^2)$ if the values $|S_i|$, $|X_i|$, $|S_i \cap S_{j+1}|$, and $|X_{i,j}|$ can be accessed in $O(1)$ time. We now show that these values can be made available in array locations $S[i]$, $X[j]$, $S[i,j]$, $X[i,j]$ by a pre-processing stage. Any interval graph recognition algorithm [3] is able to output in $O(n+m)$ time the size $X[i] = |X_i|$ of any maximal clique and $S[i] = |S_i|$ of any minimal separator, and also for any vertex x the range $[Left(x), Right(x)]$ of consecutive cliques containing x. From those values, assuming for any $1 \leqslant i \leqslant q$ $X[i,i] = |X_i|$, we have for $i+1 \leqslant j \leqslant q$, $X[i,j] = X[i,j-1] + X[j] - S[j]$. To find the values $S[i,j] = |S_i \cap S_{j+1}|$ fast, we first compute the intermediary $q \times q$-matrix M such that for $i < j$, $M[i,j] = |(S_i \cap S_j) \setminus S_{j+1}|$. Since $|S_i \cap S_j| = \sum_{h \leqslant j} |(S_i \cap S_j) \setminus S_{j+1}|$, the array $S[i,j]$ can be computed as follows:

Initialize each entry of $M[i,j]$ to 0;
For any S_i $(2 \leqslant i \leqslant q)$ and $x \in S_i$ **Do If** $Right(x) = j$ **Then** add 1 to $M[i,j]$
For $i = 2$ to q **Do** | $S[i,q] = M[i,q]$
| **For** $j = q-1$ downto i **Do** $S[i,j] = S[i,j+1] + M[i,j]$

As the sum of the sizes of the minimal separators of an interval graph is bounded by m, this preprocessing requires $O(m+n+q^2)$ time. We have shown:

For $i = 1$ to t **Do**
Compute boolean $K[i]$ by the system of equations of Theorem 4
$A[i] = T$ if $K[i] = T$ or if $\exists e \in E(T_i)$ with $A[e_1] = T$ and $A[e_2] = T$
for subtrees T_{e_1}, T_{e_2} of $T_i \setminus e$; otherwise $A[i] = F$

Fig. 3. Branchwidth of chordal graph G whose clique tree T has t subtrees is $\leq k$ iff $A[t] = T$.

Theorem 6. *Branchwidth of an interval graph $G = (V, E)$ on m edges, n vertices and $q \leq n$ maximal cliques can be computed in time $O(n + m + q^2)$.*

4.2. The general algorithm

The above interval graph algorithm can be generalized to any chordal graph. Mazoit [13] conjectured that branchwidth is computable in polynomial time for any chordal graph given with a clique tree having polynomially many subtrees. We now prove his conjecture.

For a subtree T' of a tree T we define its *connection points* as the pairs of vertices $a_1b_1, a_2b_2, \ldots, a_pb_p$, such that a_ib_i is an edge of T with $a_i \in T'$ and $b_i \in T \setminus T'$. Assume that the set of subtrees of a clique tree T_G of chordal graph G are $T_1, T_2, \ldots, T_t$, ordered by size. Let T_i have connection points $a_1b_1, a_2b_2, \ldots, a_pb_p$. Define the *connection separators* of T_i to be $S_j = X_{a_j} \cap X_{b_j}$ for $1 \leq j \leq p$, where X_{a_j}, X_{b_j} are the maximal cliques of G corresponding to tree nodes a_j, b_j. Define $K[i]$ to be True if $V(T_i)$ has a k-troika respecting the connection separators $S_1, S_2, \ldots, S_p$ of T_i. The algorithm (see Fig. 3) will decide if G has branchwidth at most k:

Theorem 7. *The above algorithm computes the branchwidth of any chordal graph G.*

Proof. Let G_i be the graph induced by $\bigcup_{j \in V(T_i)} X_j$ with p extra dummy vertices adjacent to each of the p connection separators $S_1, S_2, \ldots, S_p$ of T_i. We prove by induction on the size of the subtrees that, for $1 \leqslant i \leqslant t$, $A[i] = True$ iff $bw(G_i) \leq k$. By Theorem 4 $K[i]$ is True iff the set $V(T_i)$ has a $K[i]$-troika respecting its connection separators. Thus, $A[i]$ is certainly correct if T_i has no edge. Assume $A[i]$ correct for all subtrees on f edges. For some T_i on $f + 1$ edges, if G_i has branchwidth at most k then some merged supergraph of G_i is a k-good chordal graph, by Lemma 4. Either this merged supergraph has $V(G_i)$ as one big clique, in which case $K[i]$ is True, or there is an edge e of T_i such that for the two subtrees T_{e_1} and T_{e_2} of $T_i \setminus e$ we have $bw(G_{e_1}) \leq k$ and $bw(G_{e_2}) \leq k$. Since T_{e_1} and T_{e_2} have at most f edges each, this is correctly recorded by $A[e_1]$ and $A[e_2]$. □

Theorem 8. *For a chordal graph G given with a clique tree T_G having a polynomial number t of subtrees, the above algorithm will, in polynomial time, decide if branchwidth of G is at most k.*

Proof. Correctness follows from Theorem 7. Now the timing. Since clique tree T_G on q nodes has a number of subtrees that is polynomial in q, then the number of connection points p for any subtree T_i must be logarithmic in $q \leq n$ (a subtree with p leaves has itself at least 2^p subtrees.) Corollary 2 tells us that we can then in time polynomial in n decide if $V(T_i)$ has a k-troika respecting its p subsets. □

Since a tree on n vertices has less than 2^n subtrees, each having less than $n/2$ connection points, the algorithm has an exponential factor $2^n3^{n/2}$ for any chordal graph and runtime $O((\sqrt{3} + \sqrt{3})^n n^{O(1)})$.

5. Conclusion

By using k-troikas and edge contractions in clique trees, we have in this paper, simplified and generalized the main result of Kloks et al. [11,12] on the branchwidth of interval graphs.

The use of k-troikas allowed the separation, in Section 3, of the purely set theoretic constraints from the graph theoretic ones, to simplify the '*non-trivial proof*' of [11,12]. The runtime of their algorithm was improved by a factor $n \log n$. We generalized the polynomial-time solvable cases from the class of chordal graphs having clique trees with two leaves, to those having clique trees with a polynomial number of subtrees, thereby also proving a conjecture of Mazoit from 2004 [13].

References

[1] H.L. Bodlaender, D.M. Thi likos, Constructive linear time algorithms for branchwidth, in: 24th International Colloquium on Automata, Languages, and Programming, ICALP, in: Lecture Notes in Computer Science, vol. 1256, 1997, pp. 627–637.
[2] H.L. Bodlaender, D.M. Thilikos, Graphs with branchwidth at most three, Journal of Algorithms 32 (1999) 167–194.
[3] K. Booth, G. Lueker, Testing of the consecutive ones property, interval graphs, and graph planarity testing using PQ-tree algorithms, Journal of Computer and System Sciences 13 (1976) 335–379.
[4] W. Cook, P.D. Seymour, Tour merging via branch-decompositions, Journal on Computing 15 (2003) 233–248.
[5] E. Demaine, F. Fomin, M. Hajiaghayi, D.M. Thilikos, Fixed-parameter algorithms for (k,r)-center in planar graphs and map graphs, in: 30th International Colloquium on Automata, Languages, and Programming, ICALP, in: Lecture Notes in Computer Science, vol. 2719, 2003, pp. 829–844.

[6] F. Fomin, F. Mazoit, I. Todinca, Computing branchwidth via efficient triangulation and blocks, in: Workshop on Graphs Theoretic Concept in Computer Science, in: Lecture Notes in Computer Science, vol. 3787, 2005, pp. 374–384.
[7] F. Fomin, D. Thilikos, Dominating sets in planar graphs: Branch-width and exponential speedup, in: 14th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, 2003, pp. 168–177.
[8] F. Fomin, D. Thilikos, A simple and fast approach for solving problems on planar graphs, in: 22nd Annual Symposium on Theoretical Aspect of Computer Science, STACS, in: Lecture Notes in Computer Science, vol. 2996, 2004, pp. 56–67.
[9] F. Fomin, D. Thilikos, Fast parameterized algorithms for graphs on surfaces: Linear kernel and exponential speedup, in: 31st International Colloquium on Automata, Languages, and Programming, ICALP, in: Lecture Notes in Computer Science, vol. 3142, 2004, pp. 581–592.
[10] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Acad. Press, 1980.
[11] T. Kloks, J. Kratochvíl, H. Müller, New branchwidth territories, in: 16th Ann. Symp. on Theoretical Aspect of Computer Science, STACS, in: Lecture Notes in Computer Science, vol. 1563, 1999, pp. 173–183.
[12] T. Kloks, J. Kratochvíl, H. Müller, Computing the branchwidth of interval graphs, Discrete Applied Mathematics 145 (2005) 266–145.
[13] F. Mazoit, A general scheme for deciding the branchwidth, Technical Report RR2004-34, LIP — École Normale Supérieure de Lyon, 2004. http://www.ens-lyon.fr/LIP/Pub/Rapports/RR/RR2004/RR2004-34.pdf.
[14] F. Mazoit, Décompositions algorithmiques des graphes, Ph.D. Thesis, LIP — École Normale Supérieure de Lyon, 2004.
[15] C. Paul, J.A. Telle, New tools and simpler algorithms for branchwidth, in: European Symposium on Algorithms, ESA, in: Lecture Notes in Computer Science, vol. 3669, 2005, pp. 379–390.
[16] N. Robertson, P.D. Seymour, Graph minors X: Obstructions to tree-decomposition, Journal on Combinatorial Theory Series B 52 (1991) 153–190.
[17] P.D. Seymour, R. Thomas, Call routing and the ratcatcher, Combinatorica 14 (2) (1994) 217–241.