New Tools and Simpler Algorithms for Branchwidth

Christophe Paul, Jan Arne Telle

New Tools and Simpler Algorithms for Branchwidth*

Christophe Paul[1] and Jan Arne Telle[2]

[1] CNRS - LIRMM, Montpellier, France
`paul@lirmm.fr`
[2] Department of Informatics, University of Bergen, Norway
`telle@ii.uib.no`

Abstract. We provide new tools, such as k-troikas and good subtree-representations, that allow us to give fast and simple algorithms computing branchwidth. We show that a graph G has branchwidth at most k if and only if it is a subgraph of a chordal graph in which every maximal clique has a k-troika respecting its minimal separators. Moreover, if G itself is chordal with clique tree T then such a chordal supergraph exists having clique tree a minor of T. We use these tools to 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 F. Mazoit [13] by showing that branchwidth is polynomial on 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 [14]. 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 [6,5,4,7,8] to give e.g. good heuristics for the travelling 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 that branchwidth. Efficient algorithms computing optimal branch-decompositions, like we give in this paper, could therefore be the crucial factor that can make or break the application.

The study of branchwidth has not enjoyed the rich toolbox that treewidth has with its connections to k-trees, chordal graphs of maximum clique size, intersection graphs of subtrees of a tree etc. We try to rectify this in the current paper, by introducing various new tools like k-troikas, k-good chordal graphs and good subtree representations, whose definitions will follow later. To give an example

---

* Research conducted while the second author was on sabbatical at LIRMM.

using only standard terminology, we remark that using these tools we arrive at a succinct expression of the common basis of treewidth and branchwidth: For any $k \geq 2$ a graph G on vertices $v_1, v_2, ..., v_n$ has <u>branchwidth at most k</u> (treewidth at most $k - 1$) if and only if there is a cubic tree T with subtrees $T_1, T_2, ..., T_n$ such that if v_i and v_j adjacent then subtrees T_i and T_j share at least one <u>edge</u> (node) of T, and each <u>edge</u> (node) of T is shared by at most k of the subtrees (replace underlined words by the words in parenthesis.)

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 ratcather method computing branchwidth of planar graphs in polynomial time [15]. 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, Kratochvil and Müller [12,11] pushed into new territory by showing that branchwidth is NP-complete already for split graphs and bipartite graphs, with the bulk of their paper being an $O(n^3 \log n)$ algorithm for branchwidth of interval graphs with the comment that "it is somewhat surprising that this algorithm is by no means straightforward and its correctness proof requires a nontrivial proof." In contrast, using our branchwidth tools for the case of interval graphs we arrive at a straightforward $O(n^2)$ algorithm whose self-contained correcteness proof is easy to follow. In fact, our algorithm has runtime $O(m + n + q^2)$ for an interval graph on m edges, n vertices and q maximal cliques. In a recent investigation 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 a structural property stating that branchwidth of a chordal graph with clique tree T can be found by considering chordal supergraphs whose clique tree is a minor of T.

In Section 2 we give some standard definitions. In Section 3 we use subtree-representations to characterize graphs of branchwidth k as subgraphs of chordal graphs. In Section 4 we study the central new concept of k-troikas in a purely set-theoretic setting. In Section 5 we give 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

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 $E(G) \subseteq E(H)$. 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. 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. [1]

3   Good Subtree-Representations

Definition 1. *A* subtree-representation $R = (T, \{T_1, T_2, ..., T_n\})$ *is a pair where T is a tree with vertices of degree at most three and $T_1, T_2, ..., T_n$ are subtrees of T. Its* edge intersection graph $EI(R)$ *has vertex set $\{v_1, v_2, ..., v_n\}$ and edge set $\{v_i v_j : T_i$ and T_j share an edge of $T\}$, while its* vertex intersection graph $VI(R)$ *has the same vertex set but edge set $\{v_i v_j : T_i$ and T_j share a node of $T\}$. For a node u of T, we call the set of vertices $X_u = \{v_i : T_i$ contains $u\}$ the* bag *of u, and $\{X_u : u \in V(T)\}$ the* bags *of R.*

With the above terminology we can easily move between the view of a subtree-representation R as a tree T with a set of subtrees $\{T_1, T_2, ..., T_n\}$ or as a tree T with a set of bags $\{X_u : u \in V(T)\}$. When manipulating the latter we must simply ensure that for any vertex in $EI(R)$ the set of bags containing that vertex corresponds to a set of nodes of T inducing a subtree, i.e. a connected subgraph.

Definition 2. *The* edge-weight *of subtree-representation $R = (T, \{T_1, ..., T_n\})$ is the maximum, over all edges uv of T, of the number of subtrees in $\{T_1, ..., T_n\}$ that contain edge uv. R is a* good *subtree-representation if $EI(R) = VI(R)$.*

We are in this paper only interested in the edge intersection graphs of subtree-representations having bounded edge-weight k. We start by showing that we can restrict ourselves to good subtree-representations if we want.

Lemma 1. *For any subtree-representation R of edge-weight k there exists a good subtree-representation R' of edge-weight k with $EI(R) = EI(R') = VI(R')$.*

---

[1] 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.*

Lemma 2. *A graph G has branchwidth at most k $\Leftrightarrow$ there is a good subtree-representation R of edge-weight at most k with $G \subseteq EI(R)$.*

Proof: $\Rightarrow$: Take a branch-decomposition (T, μ) of G of width k, i.e. $|mid(e)| \leq k$ for each $e \in E(T)$. We construct a subtree-representation $R = (T', S)$ of edge-weight k with $G \subseteq EI(R)$. T' is constructed from T by for each leaf l of T adding a new leaf l' and making it adjacent to l. For vertex $a \in V(G)$ consider the smallest spanning subtree of T containing all leaves of T that are mapped by μ to an edge incident with a. The subtree T_a will be this subtree augmented by leaf l' for each leaf l of T that it contains. This completes the description of $R = (T', \{T_a : a \in V(G)\}$. For any two adjacent vertices $\{a, b\}$ of G we have $\mu^{-1}(l) = \{a, b\}$ for some leaf l of T, and thus the subtrees corresponding to a and b share the edge ll' of T' which implies that $G \subseteq EI(R)$. If vertex a has subtree T_a containing edge e of T, then there are edges incident with a mapped to leaves in both subtrees of T arising from deleting the edge e, and thus $a \in mid(e)$. But this means that the edge-weight of R is at most k. If R is not good then we can make it good by applying Lemma 1.

$\Leftarrow$: Let $R = (T, S)$ be a good subtree-representation R of edge-weight at most k with $G \subseteq EI(R)$. We construct a branch-decomposition (T', μ) of G with width k. Associate each edge ab of G with an edge e of T such that the subtrees T_a and T_b corresponding to a and b both contain e. Subdivide the tree edge e by as many new nodes as there are edges of G associated to e, thus creating for each edge ab associated to e a new tree node e_{ab}. Furthermore, add a new leaf node l_{ab}, make it adjacent to e_{ab} and set $\mu(ab) = l_{ab}$. Let T''' be the tree we have constructed so far. It contains T as a minor. Consider the smallest spanning subtree T'' of T''' having the set of leaves $\{l_{ab} : ab \in E(G)\}$. Iteratively contract edges of T'' incident to a vertex of degree two until all inner vertices have degree three. The resulting tree is T'. Note that as we constructed T' from T in stages we could at each stage have updated the subtree T_a corresponding to vertex a to a new subtree T_a' so that we would still have a subtree-representation $R' = (T', S')$ with $G \subseteq EI(R')$. For example, T_a' should contain every 'subdivision node' on a tree edge f if T_a contained f, it should contain l_{ab} for any edge ab incident with a, and it should naturally shrink if it contained a removed leaf or contracted edge. Moreover, (T', S') has edge-weight at most k since never during this process did we increase the edge-weight beyond what it was. T' has nodes of degree one and three only and μ is a bijection between its leaves and the edges of G so (T', μ) is a branch-decomposition of G. It remains to show that it has width k, i.e. that for any edge e of T' we have $|mid(e)| \leq k$. We claim that $mid(e) \subseteq \{a : T_a' \text{ contains edge } e\}$. Consider $a \in mid(e)$. There must exist two leaves l_{ab}, l_{ac} of T', one in each of the two subtrees of $T' \backslash e$, such that $a \in \mu^{-1}(l_{ab})$ and $a \in \mu^{-1}(l_{ac})$. Since the subtree T_a' of a contains both l_{ab} and l_{ac} it must also contain e. $\qquad \square$

We introduce the concept of k-troikas[2] which is a central tool in our investigation of branchwidth.

---

[2] 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 3. *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 $S_1, S_2, ..., S_q$ if any $S_i, 1 \leq i \leq q$ is contained in at least one of A, B or C.*

Definition 4. *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 1. *A graph G has branchwidth at most $k \Leftrightarrow G$ is subgraph of a k-good chordal graph*
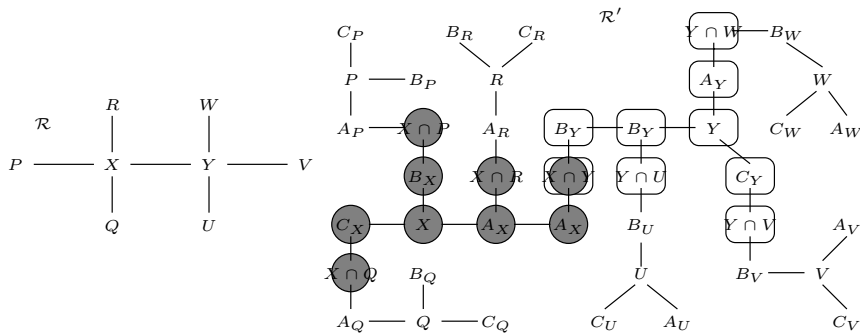
Fig. 1. On right a clique tree of a k-good chordal graph H with k-troika of any maximal clique M being (A_M, B_M, C_M). On left, the constructed subtree-representation R' of edge-weight k such that $H \subseteq EI(R')$. The square nodes correspond to the ternary subtree associated with clique Y and the grey nodes to the ternary subtree associated to clique X. Both ternary subtrees share the leaf $X \cap Y$ where they connect.

Proof: $\Rightarrow$: By Lemma 2 there exists a good subtree-representation R of edge-weight k with $G \subseteq EI(R) = VI(R)$. Since $VI(R)$ is a vertex intersection graph of subtrees of a tree it is a chordal graph [9], and $H = EI(R) = VI(R)$ will indeed be our chordal graph H having G as a subgraph. By the Helly property of (vertex) intersection of subtrees of a tree, every maximal clique of H is a bag X_u for some node u of the tree. If $|X_u| \leq k$ then it clearly has a k-troika respecting any subset, so let us assume $|X_u| > k$. Since any pair a, b of nodes from X_u is adjacent in H, we must have $\{a, b\}$ contained also in one of the neighboring bags. Let the intersection of X_u and the bags of its three neighbors be A, B and C. This means that any two of A, B, C must have union X_u since if for example $a \in X_u$ but $a \notin A \cup B$ then we would be forced to have $C = X_u$, since C would have to contain a and all its neighbors in X_u contradicting the fact that R has edge-weight k. Any minimal separator S of the chordal graph H is the intersection of two maximal cliques corresponding to two bags X_u, X_v. If we assume $A = X_u \cap X_w$, for w the neighbor of u on the path from u to v in T, then we have $S = X_u \cap X_v \subseteq A$ since the subtree corresponding to a vertex $a \in (X_u \cap X_v) \setminus A$ would be disconnected.

$\Leftarrow$: Consider any clique tree of the k-good chordal graph H containing G. In fact this can be viewed as a pair $R = (T, S)$ just as our subtree-representations with $H = VI(R)$ and every bag inducing a maximal clique of H, except that nodes of T can have degree larger than 3. We construct from this a subtree-representation $R' = (T', S')$ of edge-weight k with $G \subseteq H \subseteq EI(R')$ which by Lemma 2 and Lemma 1 will imply that G has branchwidth at most k. Let X be a maximal clique whose node in T has q neighbors corresponding to maximal cliques $Z_1, Z_2, ..., Z_q$, and let (A, B, C) be the k-troika of X respecting minimal separators $X \cap Z_1, ..., X \cap Z_q$. This means there exists a partition P_A, P_B, P_C of $\{1, 2, ..., q\}$ such that $X \cap Z_i \subseteq A$ for $i \in P_A$, $X \cap Z_i \subseteq B$ for $i \in P_B$, $X \cap Z_i \subseteq C$ for $i \in P_C$. For maximal clique X we construct a ternary subtree as follows: We have a central node with bag X adjacent to three paths: one path with $\max\{1, |P_A|\}$ bags A, one path with $\max\{1, |P_B|\}$ bags B and one with $\max\{1, |P_C|\}$ bags C. For each $i \in \{1, 2, ..., q\}$ we have a leaf-node with bag $X \cap Z_i$ as neighbor of a node on these paths, e.g. if $i \in P_A$ the leaf-node should be the neighbor of a node with bag A, if $i \in P_B$ then B, and if $i \in P_C$ then C, such that q of the nodes on the 3 paths get one leaf each. (see Figure 1). Construct such a ternary subtree for each maximal clique X, i.e. for each node of T. Then, for each pair of maximal cliques X, Y that are bags of two neighboring nodes in T we identify the following two leaves into a single node: $X \cap Y$ in the subtree constructed for X and $Y \cap X$ in the subtree constructed for Y. The resulting tree T' has no node of degree more than three and together with bags as indicated it forms the subtree-representation $R' = (T', S')$. R' has edge-weight at most k since any part of a k-troika has size at most k. We show that $H \subseteq EI(R')$. For any edge $ab \in E(H)$ we have $\{a, b\} \subseteq X$ for some maximal clique X. The k-troika (A, B, C) of X has the property that any vertex $a \in X$ must be in two out of A, B, C, so that we must have $\{a, b\}$ contained in one of A, B or C. Thus the edge ab is in $EI(R')$ and $H \subseteq EI(R')$. $\qquad \square$

4 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 1 be useful for designing algorithms computing branchwidth of graphs.

Observation 1. *If X has a k-troika respecting $S_1, S_2, ..., S_q$ 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 $|S_1 \cup S_2 \cup ... \cup S_q| \leq k$ then G has a

k-troika respecting $S_1, S_2, ..., S_q$ precisely when $|X| \leq \lfloor 3k/2 \rfloor$ since we may as well view the union of all the subsets as a single subset.

4.1 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. This motivates the following definition.

Definition 5. *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 $S_1 \cap S_2 \subseteq T_3$, $S_1 \subseteq T_1 \cap T_3$, and $S_2 \subseteq T_2 \cap T_3$.*

Observation 2. *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 iff the latter does (assuming $|S_1 \cup S_2| > k$ as discussed above).*

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. In Observation 1 we gave some obviously necessary conditions on $|X|, |S_1|, |S_2|$. What other necessary conditions do we have? Note that if $|X| = 3k/2$ and k is even then only a 'balanced' k-tripartition with each partition class having $k/2$ vertices will do. Since we must have $S_1 \cap S_2 \subseteq T_3$ the case where $|S_1 \cap S_2| > k/2$ therefore implies a stronger size restriction on X. The best we could hope for 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:

Observation 3. *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 2. *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 $|X| \leq 2k - |S_1 \cap S_2|$*

Corollary 1. *The smallest k such that X has a k-troika respecting S_1, S_2 is $\max\{|S_1|, |S_2|, \lceil 2|X|/3 \rceil, \min\{|S_1 \cup S_2|, (\lceil |X| + |S_1 \cap S_2| \rceil)/2\}\}$ 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.

4.2 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 Figure 2).
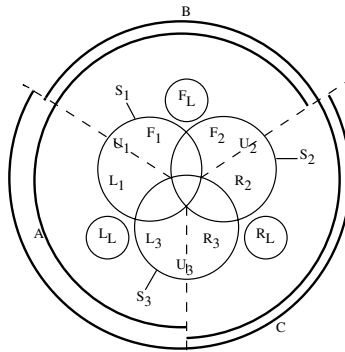
Fig. 2. Venn diagram of a set X consisting of the 6 circles $S_1, S_2, S_3, F_L, L_L, R_L$. If X has k-troika (A, B, C) respecting S_1, S_2, S_3 then we may as well require $S_1 \cap S_2 \cap S_3 = A \cap B \cap C$. The sets A, B, C are illustrated using the dotted lines at 2, 6 and 10 o'clock, e.g. A contains elements between 6 and 2 o'clock. Elements belonging to only one of the S_i sets are named U_i and further partitioned in two parts by the dotted lines.

Lemma 3. *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:*

$$a \leq |U_1|; b \leq |U_2|; c \leq |U_3|; d + e \leq |L|$$
$$|S_3| + |U_2| + a - b + d + e \leq k$$
$$|S_1| + |U_3| + |L| + b - c - e \leq k$$
$$|S_2| + |U_1| + |L| - a + c - d \leq k$$

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 4. *X has a k-troika respecting $S_1, S_2, S_3 \Leftrightarrow$ it has one satisfying the conditions of Lemma 3 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 Lemma 2.*

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 3. *X has a k-troika respecting $S_1, S_2, ..., S_q \Leftrightarrow$ there exists a partition of $\{1, 2, ..., q\}$ into three classes P_1, P_2, P_3 such that by Lemma 4 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 $S_1, S_2, ..., S_q$.*

5 Algorithms Computing Branchwidth

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 $\{1, 2, ..., q\}$ such that node i corresponds to maximal clique X_i. Mazoit [13] conjectured that branchwidth is computable in polynomial-time for any chordal graph given with a clique tree having polynomially many subtrees. We will prove his conjecture, but along the way we also give a fast algorithm for the case of interval graphs, i.e. when the clique tree is a path. We first define a merged supergraph of G which is obtained by taking certain sets of maximal cliques that are connected in T_G and merging each set into a larger clique.

Definition 6. *H is a merged supergraph of G if there exists a partition of T_G into subtrees $\{H_1 \ldots H_h\}$ (each node $j \in V(T_G)$ belongs to one and only one subtree H_i) such that the set of maximal cliques in H is: $\{X_i' = \cup_{j \in H_i} X_j\}$ ($1 \leqslant i \leqslant h$).*

It is straightforward to see that a merged supergraph H of a chordal graph G is chordal with clique-tree T_H built by making maximal cliques X_i' and X_j' adjacent iff H_i and H_j contains two adjacent nodes of T_G, in other words T_H is a minor of T_G. We first show that to find the branchwidth k of G it suffices to search for k-good chordal graphs among the merged supergraphs of G.

Lemma 5. *Let G be a chordal graph of $bw(G) = k$ and let H be a k-good chordal supergraph of G. Let X be a maximal clique of G whose neighboring maximal cliques in T_G are $X_1, X_2 \ldots X_l$. If X does not have a k-troika respecting the minimal separators in X, then there exists X_i ($1 \leqslant i \leqslant l$) such that $X_i \cup X$ is a clique in H.*

Lemma 6. *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 1 the existence of a k-good chordal graph H that is a merged supergraph of G implies that $bw(G) \leqslant k$.

$\Rightarrow$: By induction on the number q of maximal cliques of G. If G has at most 2 maximal cliques, then Lemma 5 establishes the claim. Assume by induction that the property holds for any chordal graph of branchwidth k having $q \geq 2$ maximal cliques. If G is not a k-good chordal graph, then it has a maximal clique X which does not have a k-troika respecting the minimal separators $X_1 \cap$

$X, X_2 \cap X, \ldots, X_l \cap X$, where $X_1 \ldots X_l$ are the neighbors of X in the clique tree T_G. Since G has branchwidth k it has some k-good chordal supergraph in which, by Lemma 5, some neighbor X_j $(1 \le j \le l)$ has been merged with X into a bigger clique. But then consider the merged supergraph of G arising from merging exactly X and X_j into one clique. It has $q - 1$ maximal cliques and by the induction hypothesis there is a k-good chordal graph H which is a merged supergraph of G' and therefore also of G. $\qquad\square$

5.1   Branchwidth of Interval Graphs

A graph is an interval graph iff it enjoys a *consecutive clique arrangement* (cca) that is an ordering of its maximal cliques $\mathcal{C} = (X_1, \ldots X_q)$ such that for any vertex x, the maximal cliques containing x occur consecutively. 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 $1 < i \le q$, the set $S_i = X_{i-1} \cap X_i$ is a minimal separator. Let $S_1 = S_{q+1} = \emptyset$ be dummy separators. Let us denote by $X_{i,j} = \cup_{i \le g \le j} X_g$ $(1 \le i \le j \le q)$ a merged set of consecutive cliques.

Given a cca $\mathcal{C}_G = (X_1 \ldots X_q)$ of an interval graph G, a merged supergaph H of G has caa $\mathcal{C}_H = (X'_1 \ldots X'_h)$ with $h \le q$ such that for any $1 \le i \le h$, $X'_i = X_{l_i, r_i}$ with $l_1 = 1$, $l_i = r_{i-1} + 1$ for $i > 1$ and $r_h = q$. Note that a merged supergraph of an interval graph is also an interval graph.

Our algorithm first computes for each pair $1 \le i \le j \le q$ 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}. Then by simple dynamic programming it computes the best way of merging various such sets into a merged supergraph, see Figure 3. Incrementally, in step j, we optimize over the possible cutoff points $1 \le i \le j$ that define the 'rightmost' merged set of cliques $X_{i,j}$. We prove correctness before considering the running time.

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

Fig. 3. Computation of $bw(G) = A[q]$ for interval graph G

Theorem 4. *The computed value $A[q]$ is the branchwidth of interval graph G.*

Proof: Let us prove by induction that, for $1 \le i \le q$, $A[i] = bw(G_i)$ 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 1 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 5 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 Lemma 6, 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, $bw(G) = bw(G_q)$. $\qquad\qquad\square$

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 pre-processing stage. Any interval graph recognition algorithm [3] is able to ouput 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:

Theorem 5. *Branchwidth of an interval graph $G = (V, E)$ on m edges, n vertices and $q \leq n$ maximal cliques can be computed in time $O(n + m + q^2)$.*

5.2   Clique Trees with Polynomial Number of Subtrees

For a subtree T' of a tree T we define its *connection points* as the pairs of vertices $a_1 b_1, a_2 b_2, ..., a_p b_p$ such that $a_i b_i$ is an edge of T with $a_i \in T'$ and $b_i \in T \setminus T'$. Assume clique tree T_G of chordal graph G has a polynomial number of subtrees $T_1, T_2, ..., T_t$, ordered by size. Let T_i have connection points $a_1 b_1, a_2 b_2, ..., a_p b_p$. 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 $S_1, S_2, ..., S_p$ of T_i. The following algorithm will in polynomial time decide if G has branchwidth at most k:

Theorem 6. *For a chordal graph G given with a clique tree having a polynomial number t of subtrees the above algorithm will in polynomial time decide if branchwidth of G is at most k.*

For $i = 1$ to t **Do** Compute boolean $K[i]$ by the system of equations of Theorem 3
$A[i] = T$ 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. 4. Branchwidth of $G \leq k$ iff $A[t] = T$

References

1. H.L. Bodlaender and D.M. Thilikos. Constructive linear time algorithms for branchwidth. In *24th International Colloquium on Automata, Languages, and Programming (ICALP)*, Vol. 1256 of *Lecture Notes in Computer Science*, p. 627–637, 1997.
2. H.L. Bodlaender and D.M. Thilikos. Graphs with branchwidth at most three. *Journal of Algorithms*, 32:167–194, 1999.
3. K. Booth and G. Lueker. Testing of the consecutive ones property, interval graphs, and graph planarity testing using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13:335–379, 1976.
4. W. Cook and P.D. Seymour. Tour merging via branch-decompositions. *Journal on Computing*, 15:233–248, 2003.
5. E. Demaine, F. Fomin, M. Hajiaghayi, and 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)*. Vol. 2719 of *Lecture Notes in Computer Science*, p. 829–844, 2003.
6. F. Fomin and D. Thilikos. Dominating sets in planar graphs: Branch-width and exponential speedup. In *14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, p. 168–177, 2003.
7. F. Fomin and D. Thilikos. A simple and fast approach for solving problems on planar graphs. In *22nd Annual Symposium on Theoretical Aspect of Computer Science (STACS)* Vol. 2996 of *Lecture Notes in Computer Science*, p. 56-67, 2004.
8. F. Fomin and D. Thilikos. Fast parameterized algorithms for graphs on surfaces: Linear kernel and exponential speedup In *31st International Colloquium on Automata, Languages, and Programming (ICALP)*, Vol. 3142 of *Lecture Notes in Computer Science*, p. 581-592, 2004.
9. F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory Series B*, 16:47–56, 1974.
10. M.C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Acad. Press, 1980.
11. T. Kloks, J. Kratochvil, and H. Müller. New branchwidth territories. In *16th Ann. Symp. on Theoretical Aspect of Computer Science (STACS)* Vol. 1563 of *Lecture Notes in Computer Science*, p. 173–183, 1999.
12. T. Kloks, J. Kratochvil, and H. Müller. Computing the branchwidth of interval graphs. *Discrete Applied Mathematics* 145:266-145, 2005.
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. N. Robertson and P.D. Seymour. Graph minors X: Obstructions to tree-decomposition. *Journal on Combinatorial Theory Series B*, 52:153–190, 1991.
15. P.D. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994.