On the Complexity of Computing Treebreadth

Guillaume Ducoffe, Sylvain Legay, Nicolas Nisse

On the complexity of computing treebreadth ⋆

Guillaume Ducoffe[1,2], Sylvain Legay[3], and Nicolas Nisse[2,1]

[1] Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, 06900 Sophia Antipolis, France
[2] Inria, France
[3] LRI, Univ Paris Sud, Université Paris-Saclay, 91405 Orsay, France

Abstract. During the last decade, metric properties of the *bags* of tree-decompositions of graphs have been studied. Roughly, the *length* and the *breadth* of a tree-decomposition are the maximum diameter and radius of its bags respectively. The *treelength* and the *treebreadth* of a graph are the minimum length and breadth of its tree-decompositions respectively. *Pathlength* and *pathbreadth* are defined similarly for path-decompositions. In this paper, we answer open questions of [Dragan and Köhler, Algorithmica 2014] and [Dragan, Köhler and Leitert, SWAT 2014] about the computational complexity of treebreadth, pathbreadth and pathlength. Namely, we prove that computing these graph invariants is NP-hard. We further investigate graphs with treebreadth one, i.e., graphs that admit a tree-decomposition where each bag has a dominating vertex. We show that it is NP-complete to decide whether a graph belongs to this class. We then prove some structural properties of such graphs which allows us to design polynomial-time algorithms to decide whether a bipartite graph, resp., a planar graph, has treebreadth one.

1 Introduction

Tree-decompositions [20] aim at decomposing graphs into pieces, called *bags*, organized in a tree-like manner (formal definitions are postponed to Section 1.3). Roughly, the *width* of a tree-decomposition is the maximum size of its bags. A lot of work has been dedicated to compute tree-decompositions with small width since such decompositions can be efficiently exploited for algorithmic purposes [4]. Computing the corresponding graph invariant, the *treewidth* of a graph G (i.e., the minimum width among all tree-decompositions of G), is NP-hard [2] and no constant-approximation algorithm is likely to exist [22]. Moreover, real-life networks generally have a large treewidth [11]. These drawbacks motivated the study of other optimization criteria for tree-decompositions.

In particular, the metric properties of the bags have been studied. Roughly, the *length* and the *breadth* of a tree-decomposition are the maximum diameter and radius of its bags respectively. The corresponding graph parameters are the *treelength* [13] and the *treebreadth* [14] respectively. Recent studies suggest that some classes of real-life networks – including biological networks and social networks – have bounded treebreadth [1]. This metric tree-likeness can be exploited

⋆ This work is partially supported by ANR project Stint under reference ANR-13-BS02-0007 and ANR program "Investments for the Future" under reference ANR-11-LABX-0031-01.

in algorithms. For instance, bounded treebreadth graphs admit a PTAS for the Traveling Salesman problem [18]. They also admit compact distance labeling schemes [12]. Furthermore, the diameter and the radius of bounded treebreadth graphs can be approximated up to an additive constant in linear time [9]. In contrast to the above result, we emphasize that under classical complexity assumptions the diameter of general graphs *cannot* be approximated up to an additive constant in subquadratic time, that is prohibitive for large graphs [8].

On the computational side, it is known that computing the treelength is NP-hard [19]. However, contrary to the treewidth, there exists a 3-approximation algorithm for computing the treelength [13]. In [14], a 3-approximation algorithm for computing the treebreadth is presented but the computational complexity of this problem is left open. Note that, because treelength and treebreadth differ by at most a factor 2 [14], any polynomial-time algorithm for computing the treebreadth, or an α-approximation algorithm for some $\alpha < 3/2$, would improve the 3-approximation algorithm for treelength [13].

A *path-decomposition* of a graph is a tree-decomposition where the bags are organized according to a path structure. Treelength and treebreadth have their "path counterpart", namely the *pathlength* and the *pathbreadth*. In [15], they have been shown to be useful in the design of approximation algorithms for bandwidth and line-distortion. A 2-approximation (resp., a 3-approximation) algorithm is given for computing the pathlength (resp., the pathbreadth) but the computational complexity of both problems is left open.

The main contributions of this paper are to answer the open problems of [14] and [15]. Namely, we prove that computing the treebreadth, pathlength and pathbreadth of graphs are all NP-hard problems.

1.1 Related work.

In contrast with treewidth [5], deciding whether a graph has treelength at most k is NP-complete for every fixed $k \geq 2$ [19]. However, the reduction used for treelength goes through weighted graphs and then goes back to unweighted graphs using rather elegant gadgets. It does not seem to us these gadgets can be easily generalized in order to apply to the treebreadth.

Relationship between treewidth and treelength (and so, treebreadth) has been investigated in [10]. The two parameters are uncomparable in general graphs. For instance, cycles have treewidth at most two but treelength $\lceil n/3 \rceil$, while cliques have treewidth $n-1$ but treelength equal to one [13]. However, they differ by at most a constant ratio in the graphs with bounded genus and bounded isometric cycles [10]. Hence we are also motivated in this work to better understand the structure of tree-decompositions with small width for bounded genus graphs, and to improve their computation.

Recently, the minimum eccentricity shortest-path problem – close to the problem of computing the pathlength and pathbreadth – has been proved NP-hard [16]. Let us point out that for every fixed k, it can be decided in polynomial time whether a graph admits a shortest-path with eccentricity at most k [16]. Our results will show the situation is different for pathlength and pathbreadth.

1.2 Our contributions.

On the negative side, we prove in Section 2 that computing the treebreadth is NP-hard. More precisely, we first prove that recognizing graphs with treebreadth one is NP-complete. The latter may be a bit surprising since in comparison, graphs with treelength one are exactly the chordal graphs [19], and so, they can be recognized in linear time. Our reduction has distant similarities with the one for treelength. However, it does not need any detour through weighted graphs. Then, we show that the problem of deciding whether a graph has treebreadth one is polynomially equivalent to the problem of deciding whether a graph has treebreadth at most k, for every fixed $k \geq 1$.

Next, we show that deciding if a graph has pathlength at most 2 is NP-hard even in the class of graphs with pathlength at most 3. We also show that deciding if a graph has pathbreadth at most 1 is NP-hard even in the class of graphs with pathbreadth at most 2. Hence, for any $\epsilon > 0$, the pathlength and the pathbreadth cannot be approximated within a factor $\frac{3}{2} - \epsilon$ and $2 - \epsilon$ respectively unless $P = NP$.

On the positive side, we present polynomial-time algorithms for deciding whether a graph has treebreadth at most one, in the class of bipartite graphs and in the class of planar graphs. Precisely, we prove that a bipartite graph has treebreadth one if and only if it can be clique-decomposed in *tree-convex* bipartite graphs [21]. Furthermore, while the planar graphs of treebreadth one are quite specific (in particular, we prove that they have treewidth at most 4), the algorithm is intricate and relies on structural properties of graphs with treebreadth one.

Due to lack of space, several proofs are only sketched or even omitted. They can be found in our technical report [17].

1.3 Definitions and notations

Graphs in this study are finite, simple, connected and unweighted. Given a graph $G = (V, E)$, the set $N_G(v)$ denotes the set of neighbors of $v \in V$ in G. Furthermore, let $N_G[v] = N_G(v) \cup \{v\}$. The distance $dist_G(u, v)$ between two vertices $u, v \in V$ in G is the minimum length (number of edges) of a path between u and v in G. We will omit the subscript when no ambiguity occurs.

A *tree-decomposition* $(T, \mathcal{X})$ of G is a pair consisting of a tree T and of a family $\mathcal{X} = (X_t)_{t \in V(T)}$ of subsets of V indexed by the nodes of T and satisfying:

- $\bigcup_{t \in V(T)} X_t = V$;
- for any edge $e = \{u, v\} \in E$, there exists $t \in V(T)$ such that $u, v \in X_t$;
- for any $v \in V$, $\{t \in V(T) \mid v \in X_t\}$ induces a subtree, denoted by T_v, of T.

The sets X_t are called *the bags* of the decomposition. For any $t \in V(T)$, the *diameter* of the bag X_t equals $\max_{v,w \in X_t} dist_G(v, w)$. We emphasize that the distance is the one in G (not in $G[X_t]$). The *radius* of X_t equals $\min_{v \in V} \max_{w \in X_t} dist_G(v, w)$. We point out that the vertex v in previous definition does not necessarily belong to X_t. The *length* of $(T, \mathcal{X})$ is the maximum diameter of its bags, while the *breadth* of $(T, \mathcal{X})$ is the maximum radius of its bags.

The *treelength* and the *treebreadth* of G, respectively denoted by $tl(G)$ and $tb(G)$, are the minimum length and breadth of its tree-decompositions, respectively. Pathlength and pathbreadth are defined similarly in the case of path decompositions, that is, when T is a path. It has been observed in [14, 15] that the four above parameters are contraction-closed invariants.

A tree-decomposition is called *reduced* if no bag is included in another one. Starting from any tree-decomposition, a reduced tree-decomposition can be obtained in polynomial time by contracting any two adjacent bags with one contained in the other until it is no more possible to do that. Note that such a process does not modify the width, the length nor the breadth of the decomposition.

In the following we will make use of the well-known *Helly property* in our proofs: any family of pairwise intersecting subtrees in a tree has a nonempty intersection.

2 Hardness of treebreadth, pathlength and pathbreadth

The main result of this section is the NP-completeness of deciding whether $tb(G) \leq k$, for any fixed $k \geq 1$. We first prove that the problem is NP-complete for $k = 1$. Then, we show that the problem of deciding the treebreadth of a graph is polynomially equivalent to the problem of recognizing graphs with treebreadth one. Using similar techniques, we prove that computing pathlength, resp., pathbreadth, is NP-hard.

We start by a structural result on graphs with treebreadth one which will be a key lemma used throughout the paper. A tree-decomposition $(T, \mathcal{X})$ of a graph is a *star-decomposition* if for each $t \in V(T)$, $X_t \subseteq N[v]$ for some $v \in X_t$. That is, star-decompositions are similar to decompositions of breadth one, but the dominator of each bag has to belong to the bag itself. Lemma 1 shows that both definitions are actually equivalent.

Lemma 1. *For any graph G with $tb(G) \leq 1$, every* reduced *tree-decomposition of G of breadth one is a star-decomposition.*

Proof. Let $(T, \mathcal{X})$ be any reduced tree-decomposition of G of breadth one. We will prove it is a star-decomposition. To prove it, let $X_t \in \mathcal{X}$ be arbitrary and let $v \in V$ be such that $\max_{w \in X_t} dist_G(v, w) = 1$, which exists because X_t has radius one. We now show that $v \in X_t$. Indeed, since the subtree T_v and the subtrees $T_w, w \in X_t$, pairwise intersect, then it comes by the Helly Property that $T_v \cap \left(\bigcap_{w \in X_t} T_w\right) \neq \emptyset$ *i.e.*, there is some bag containing $\{v\} \cup X_t$. As a result, we have that $v \in X_t$ because $(T, \mathcal{X})$ is a reduced tree-decomposition. The latter implies that $(T, \mathcal{X})$ is a star-decomposition because X_t is arbitrary. □

We then show the main result of this section.

Theorem 1. *Deciding whether a graph has treebreadth one is NP-complete.*

In order to prove Theorem 1, we reduce the following particular instance of Chordal Sandwich (proved to be NP-hard in [6]) to our problem. In [19], the author also proposed a reduction from Chordal Sandwich in order to prove that computing treelength is NP-hard. However, we will need different gadgets than in [19], and we will need different arguments to prove correctness of the reduction.

*Problem 1 (*Chordal Sandwich with $\overline{nK_2}$*).*

Input: graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $E_1 \subseteq E_2$, $|V|$ is even and the complementary $\bar{G_2}$ of G_2 induces a perfect matching.
Question: Is there a chordal graph $H = (V, E)$ such that $E_1 \subseteq E \subseteq E_2$?

Perhaps surprisingly, the restriction on the structure of $\bar{G_2}$ is a key element in our reduction. Indeed, we will need the following technical lemma whose proof can be found in [17].

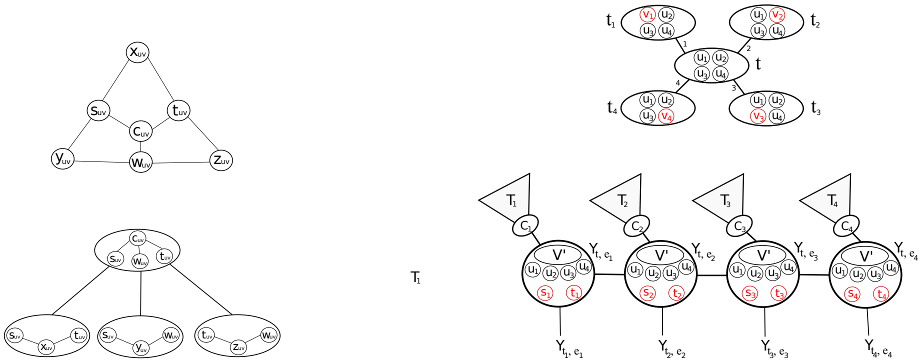
Lemma 2. *Let $G_1 = (V, E_1)$, $G_2 = (V, E_2)$ such that $E_1 \subseteq E_2$ and $\bar{G_2}$ is a perfect matching. Suppose that $\langle G_1, G_2 \rangle$ is a yes-instance of* Chordal Sandwich with $\overline{nK_2}$*.*

There exists a tree-decomposition $(T, \mathcal{X})$ of G_1 with $|\mathcal{X}| = |V|/2+1$ bags such that for every $\{u, v\} \notin E_2$, $T_u \cap T_v = \emptyset$ and there are two adjacent bags $B_u \in T_u$ and $B_v \in T_v$ such that $B_u \setminus u = B_v \setminus v$.

Proof of Theorem 1. The problem is in NP. To prove the NP-hardness, let $\langle G_1, G_2 \rangle$ be any instance of Chordal Sandwich with $\overline{nK_2}$. Let G' be the graph constructed from G_1 as follows. First, a clique V' of $2n = |V|$ vertices is added to G_1. Vertices $v \in V$ are in one-to-one correspondance with vertices $v' \in V'$. Then, for every $\{u, v\} \notin E_2$, u and v are respectively made adjacent to all vertices in $V' \setminus v'$ and $V' \setminus u'$. Finally, we add a copy of the gadget F_{uv}, depicted in Figure 1(a), and the vertices s_{uv} and t_{uv} are made adjacent to the four vertices u, v, u', v'.

We will prove $tb(G') = 1$ if and only if $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich with $\overline{nK_2}$.

In one direction, assume $tb(G') = 1$, let $(T, \mathcal{X})$ be a star-decomposition of G' (which exists by Lemma 1). We prove that the triangulation of G_1 obtained from this star-decomposition is the desired chordal sandwich. Let $H = (V, \{\{u, v\} \mid T_u \cap T_v \neq \emptyset\})$. H is a chordal graph such that $E_1 \subseteq E(H)$. To prove that $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich with $\overline{nK_2}$, it suffices to prove that $T_u \cap T_v = \emptyset$ for every $\{u, v\} \notin E_2$. We claim that it is implied by $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$. Indeed, assume $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$ and $T_u \cap T_v \neq \emptyset$. Since $s_{uv}, t_{uv} \in N(u) \cap N(v)$, $T_u, T_v, T_{s_{uv}}, T_{t_{uv}}$ pairwise intersect, there is a bag with u, v, s_{uv}, t_{uv} by the Helly property. The latter contradicts that $(T, \mathcal{X})$ is a star-decomposition because no vertex dominates the four vertices. Hence the claim is proved. So, let us prove that $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$. By contradiction, if $T_{s_{uv}} \cap T_{t_{uv}} = \emptyset$ then every bag B onto the path between $T_{s_{uv}}$ and $T_{t_{uv}}$ must contain c_{uv}, x_{uv}. Since $N[c_{uv}] \cap N[x_{uv}] = \{s_{uv}, t_{uv}\}$ and $(T, \mathcal{X})$ is a star-decomposition, it implies either $s_{uv} \in B$ and $B \subseteq N[s_{uv}]$ or $t_{uv} \in B$ and $B \subseteq N[t_{uv}]$. So, there are two



(a) Gadget F_{uv} (top) with a star-decomposition of F_{uv} (bottom).

(b) A subtree of the star-decomposition of G' (bottom) obtained from an internal bag with degree four of $(T, \mathcal{X})$ (top). Subtrees T_i are star-decompositions of the gadgets $F_{u_i v_i}$.

Fig. 1.

adjacent bags $B_s \in T_{s_{uv}}, B_t \in T_{t_{uv}}$ such that $B_s \subseteq N[s_{uv}]$ and $B_t \subseteq N[t_{uv}]$. In particular, $B_s \cap B_t$ must intersect the path (y_{uv}, w_{uv}, z_{uv}) because $y_{uv} \in N(s_{uv})$ and $z_{uv} \in N(t_{uv})$. However, $N[s_{uv}] \cap N[t_{uv}] \cap \{y_{uv}, w_{uv}, z_{uv}\} = \emptyset$, that is a contradiction. As a result, $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$ and so, $T_u \cap T_v = \emptyset$ for any $\{u,v\} \notin E_2$.

Conversely, assume that $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich with $\overline{nK_2}$. Since $\bar{G}_2$ is a perfect matching by the hypothesis, let $(T, \mathcal{X})$ be as stated in Lemma 2. We will modify $(T, \mathcal{X})$ in order to obtain a star-decomposition of G'. To do so, we will use the fact that there are $|V|/2 = n$ edges in $E(T)$ and the properties stated by Lemma 2. Indeed, this implies that there is a one-to-one mapping $\alpha : E(T) \to E(\bar{G}_2)$ between the edges of T and the non-edges of G_2. Precisely, for any edge $e = \{t, s\} \in E(T)$, let $\alpha(e) = \{u, v\} \in E(\bar{G}_2)$ be the non-edge of G_2 such that $u \in X_t, v \in X_s$ and $X_t \setminus u = X_s \setminus v$.

Intuitively, the star-decomposition $(T', \mathcal{X}')$ of G' is obtained as follows. For any $t \in V(T)$ with incident edges $e_1, \cdots, e_d$, we first replace X_t by a path-decomposition $(Y_{t,e_1}, \cdots, Y_{t,e_d})$. Then, for any edge $e = \{t, s\} \in E(T)$, an edge is added between $Y_{t,e}$ and $Y_{s,e}$. Finally, the center-bag of some star-decomposition of the gadget $F_{\alpha(e)}$ is made adjacent to $Y_{t,e}$ (see Figure 1(b) for an illustration).

More formally, let $t \in V(T)$ and $e \in E(T)$ incident to t, and let $\{u, v\} = \alpha(e)$. Let $Y_{t,e} = V' \cup X_t \cup \{s_{uv}, t_{uv}\}$ (note that $Y_{t,e}$ is dominated by $u' \in V'$). Let $e_1, \cdots, e_d$ be the edges incident to t in T, in any order. For $1 \leq i < d$, add an edge between Y_{t,e_i} and $Y_{t,e_{i+1}}$. For any edge $e = \{t, s\} \in E(T)$, add an edge between $Y_{t,e}$ and $Y_{s,e}$. Finally, add the star-decomposition $(T^e, \mathcal{X}^e)$ for the gadget $F_{\alpha(e)}$ as depicted in Figure 1(a) and add an edge between its center and $Y_{t,e}$.

The resulting $(T', \mathcal{X}')$ is a star-decomposition of G', hence $tb(G') = 1$. $\square$

We next show that computing the treebreadth is polynomially equivalent to the recognition of graphs with treebreadth one.

Lemma 3. *For every graph G, for every positive integer r, there exists a graph G'_r computable in polynomial time such that $tb(G) \leq r$ if and only if $tb(G'_r) \leq 1$.*

Proof. Let G have vertices $v_1, v_2, \ldots, v_n$, and let $r > 0$. The graph G'_r is obtained from G by adding a clique $U = \{u_1, u_2, \ldots, u_n\}$ so that for every $1 \leq i \leq n$, u_i is adjacent to all vertices in $B_G(v_i, r) = \{x \in V(G) \mid dist_G(v_i, x) \leq r\}$.

If $tb(G) \leq r$ then we claim that given a tree-decomposition $(T, \mathcal{X})$ of G with breadth at most r, one obtains a star-decomposition of G'_r by adding the clique U in every bag in $\mathcal{X}$. Indeed, for every bag $X_t \in \mathcal{X}$, by the hypothesis there is $v_i \in V(G)$ such that $\max_{x \in X_t} dist_G(v_i, x) \leq r$, hence $X_t \cup U \subseteq N_{G'_r}[u_i]$. Conversely, if $tb(G'_r) \leq 1$ then we claim that given a star-decomposition $(T', \mathcal{X}')$ of G'_r, one obtains a tree-decomposition of G with breadth at most r by removing every vertex of the clique U from every bag in $\mathcal{X}'$. Indeed, for every bag $X'_t \in \mathcal{X}'$, by the hypothesis there is $y \in X'_t$ such that $X'_t \subseteq N_{G'_r}[y]$. Furthermore, $y \in \{u_i, v_i\}$ for some $1 \leq i \leq n$, and so, since $N_{G'_r}[v_i] \subseteq N_{G'_r}[u_i]$ by construction, $X'_t \setminus U \subseteq N_{G'_r}(u_i) \setminus U = \{x \in V(G) \mid dist_G(v_i, x) \leq r\}$. $\square$

Lemma 4. *For every graph G, for every positive integer r, there exists a graph G' computable in polynomial time such that $tb(G) \leq 1$ if and only if $tb(G') \leq r$.*

Proof. For every $\{u, v\} \in E(G)$, let F^r_{uv} be obtained from F_{uv} in Figure 1(a) by adding an edge $\{s_{uv}, t_{uv}\}$ then subdividing each edge $r - 1$ times. The graph G' is obtained from G by substituting each edge $\{u, v\} \in E(G)$ with a distinct copy of F^r_{uv} then identifying u, v with s_{uv}, t_{uv}.

If $tb(G) \leq 1$ then let us modify a star-decomposition $(T, \mathcal{X})$ of G in a tree-decomposition $(T', \mathcal{X}')$ of G' of breadth at most r. Clearly, every bag in $\mathcal{X}$ has radius at most r in G'. Furthermore, let $(T^{uv}, \mathcal{X}^{uv})$ be the star-decomposition of F_{uv} in Figure 1(a), with three leaf-bags and one central bag. It can be modified in a tree-decomposition of F^r_{uv} by `i)` adding in each bag containing both end-vertices of an edge in F_{uv} the $r-1$ vertices in F^r_{uv} that result from its subdivision, and `ii)` adding a new leaf-bag with $\{u, v\}$ and the $r-1$ vertices that result from its subdivision. Finally, let $(T', \mathcal{X}')$ be obtained from $(T, \mathcal{X})$ by adding an edge between some bag in $T_u \cap T_v$ and the central bag of T^{uv} for every $\{u, v\} \in E(G)$. Since $(T', \mathcal{X}')$ has breadth r, $tb(G') \leq r$.

Conversely, if $tb(G') \leq r$ then we claim that given a tree-decomposition $(T', \mathcal{X}')$ of G' of breadth at most r, one obtains a tree-decomposition of G of breadth one by removing every vertex of $V(G') \setminus V(G)$ from the bags in $\mathcal{X}'$. Before proving the claim, observe that no vertex in $V(G') \setminus V(G)$ can be at distance at most r from three vertices in $V(G)$, and in case it is at distance at most r from two vertices $u, v \in V(G)$ then $\{u, v\} \in E(G)$. Therefore, in order to prove the claim it suffices to prove that $u = s_{uv}$ and $v = t_{uv}$ are in a common bag of $\mathcal{X}'$ for every $\{u, v\} \in E(G)$. The latter can be proved by elaborating on the same arguments as for Theorem 1. $\square$

From Lemmas 3, 4 and Theorem 1, it follows that:

Theorem 2. *For any fixed $k \geq 1$, deciding whether a graph G has treebreadth at most k is NP-complete.*

To conclude this section, we consider pathlength and pathbreadth. Due to lack of space, the proofs are postponed in [17].

Theorem 3. *For any $\epsilon > 0$, the pathlength (resp., the pathbreadth) cannot be approximated within a factor $\frac{3}{2} - \epsilon$ (resp., $2 - \epsilon$) unless $P = NP$.*

3 Graphs with treebreadth one: some polynomial cases

In this section, we investigate further the class of graphs with treebreadth one. It strictly contains chordal graphs and dually chordal graphs, well-studied graph classes in algorithmic graph theory [7]. We first show some useful lemmas that somehow state that we can restrict our study on graphs without clique-separator. Then, we show that the problem of recognizing graphs with treebreadth one can be solved in polynomial time in the class of bipartite graphs and in the class of planar graphs.

Let $G = (V, E)$ be a connected graph. Recall that a set $S \subset V$ is a *separator* if $G \setminus S$ is disconnected. It is called a *clique-separator* if S induces a complete graph. A *full component* for S is any connected component C of $G \setminus S$ such that $N(C) = S$. If C is a full component for S then we call the induced subgraph $G[C \cup S]$ a block. Finally, S is a *minimal separator* if there exist at least two full components for S.

Our objective is to prove that if a graph G has treebreadth one then so do all its blocks. In fact, we will prove a slightly more general result:

Lemma 5. *Let $G = (V, E)$, S be a separator and W be the union of some connected components of $G \setminus S$. If $tb(G) = 1$ and W contains a full component for S, then $tb(G[W \cup S]) = 1$.*

Proof. Let $(T, \mathcal{X})$ be a star-decomposition of G. We remove vertices in $V \setminus (W \cup S)$ from bags in $\mathcal{X}$, that yields a tree-decomposition $(T, \mathcal{X}')$ of $G[W \cup S]$. We will prove that $(T, \mathcal{X}')$ has breadth one (but is not necessarily a star-decomposition). Indeed, let $X'_t \in \mathcal{X}'$. By construction, $X'_t \subseteq X_t$ with $X_t \in \mathcal{X}$. Let $v \in X_t$ satisfy $X_t \subseteq N_G[v]$. If $v \in X'_t$, then we are done. Else, since for all $x \notin S \cup W, N(x) \cap (S \cup W) \subseteq S$ (because S is a separator by the hypothesis), we must have that $X_t \subseteq S$. Let $A \subseteq W$ be a full component for S, that exists by the hypothesis, let T_A be induced by the bags intersecting A. Since T_A and the subtrees $T_x, x \in X_t$ pairwise intersect — because for all $x \in X_t$, $x \in S$ and so, x has a neighbour in A —, then by the Helly property there is a bag in $\mathcal{X}$ containing X_t and intersecting A. Furthermore, any $u \in V$ dominating this bag must be either in S or in A, so, in particular there is $u \in A \cup S$ such that $X_t \subseteq N[u]$. □

The converse of Lemma 5 does not hold in general (see Fig. 2), yet there are interesting cases when it does.

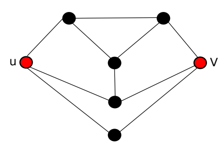


Fig. 2. $S = \{u, v\}$ separates G in two subgraphs of treebreadth 1. However, $tb(G) = 2$.

Lemma 6. *Let $G = (V, E)$ with a minimal clique-separator S and A be a full component. Then, $tb(G) = 1$ if and only if $tb(G[A \cup S]) = 1$ and $tb(G[V \setminus A]) = 1$.*

The proof of Lemma 6 is deferred to [17]. Recall that computing the clique-minimal-decomposition of a graph G takes $\mathcal{O}(nm)$-time, where m denotes the number of edges [3]. By doing so, one replaces a graph G with the maximal subgraphs of G that have no clique-separator, *a.k.a. atoms*. So, in the following we will only consider graphs without a clique-separator, *a.k.a.*, *prime graphs*.

3.1 Bipartite graphs

Bipartite graphs with treebreadth one are an interesting subclass of their own since they contain the convex bipartite graphs and the chordal bipartite graphs (*i.e.*, bipartite graphs with no induced cycle of length at least six). In this section, we present a linear-time algorithm that decides whether a prime bipartite graph has treebreadth one, and computes a corresponding decomposition if any. Since the clique-decomposition of a given bipartite graph can be computed in linear time, this proves combined with Lemma 6 that it can be decided in linear time whether a bipartite graph has treebreadth one.

More precisely, we show that prime bipartite graphs with treebreadth one coincide with *tree-convex* bipartite graphs, a generalization of convex bipartite graphs [21]. A bipartite graph is called tree-convex if it admits a tree-decomposition where the bags are the close neighbourhoods of any one side of its bipartition. By definition, tree-convex graphs have treebreadth one. The following lemma is a converse of this result.

Lemma 7. *Let $G = (V_0 \cup V_1, E)$ be a prime bipartite graph with treebreadth one. There is $(T, \mathcal{X})$ a star-decomposition of G such that either $\mathcal{X} = \{N[v_0] \mid v_0 \in V_0\}$, or $\mathcal{X} = \{N[v_1] \mid v_1 \in V_1\}$.*

Proof. Let $(T, \mathcal{X})$ be a star-decomposition of G minimizing $|\mathcal{X}|$. Suppose there is some $v_0 \in V_0$, there is $t \in V(T)$ such that $X_t \subseteq N_G[v_0]$ (the case when there is some $v_1 \in V_1$, there is $t \in V(T)$ such that $X_t \subseteq N_G[v_1]$ is symmetrical to this one). We claim that for every $t' \in V(T)$, there is $v_0' \in V_0$ such that $X_{t'} \subseteq N_G[v_0']$. By contradiction, let $v_0 \in V_0, v_1 \in V_1$, let $t, t' \in V(T)$ be such that $X_t \subseteq N_G[v_0], X_{t'} \subseteq N_G[v_1]$. By connectivity of the tree T we may assume w.l.o.g. that $\{t, t'\} \in E(T)$. Moreover, $N_G(v_0) \cap N_G(v_1) = \emptyset$ because G is bipartite. Therefore, $X_t \cap X_{t'} \subseteq \{v_0, v_1\}$, and in particular if $X_t \cap X_{t'} = \{v_0, v_1\}$ then v_0, v_1 are adjacent in G. However, by the properties of a tree-decomposition this implies that $X_t \cap X_{t'}$ is a clique-separator (either an edge or a single vertex), thus contradicting the fact that G is prime.

Let $v_0 \in V_0$ be arbitrary. We claim that there is a unique bag $X_t,\ t \in V(T)$, containing v_0. Indeed, any such bag X_t must satisfy $X_t \subseteq N_G[v_0]$, hence the subtree T_{v_0} can be contracted into a single bag $\bigcup_{t \in T_{v_0}} X_t$ without violating the property for the tree-decomposition to be a star-decomposition. As a result, the uniqueness of the bag X_t follows from the minimality of $|\mathcal{X}|$. Since X_t is unique and $X_t \subseteq N_G[v_0]$, therefore $X_t = N_G[v_0]$ and so, $\mathcal{X} = \{N[v_0] \mid v_0 \in V_0\}$. $\square$

As shown in [21], tree-convex graph recognition can be reduced to hypertree recognition, that can be done in linear time [7]. Altogether, we obtain the following characterization of bipartite graphs with treebreadth one.

Corollary 1. *A bipartite graph has treebreadth one if and only if every of its atoms is tree-convex, which can be decided in linear time.*

3.2 Planar graphs

In this section, we sketch a quadratic algorithm to recognize prime planar graphs of treebreadth one. Combined with Lemma 6, this shows that planar graphs of treebreadth one can be recognized in quadratic time. Our algorithm also allows to compute a corresponding decomposition in cubic time. Since the full analysis is lengthy, all proofs in this section are deferred to [17].

Our work in this section brings more insights on tree-decompositions with small width for planar graphs. Indeed, we prove the following.

Lemma 8. *For every planar graph G, $tb(G) \leq 1$ implies $tw(G) \leq 4$.*

The algorithm is recursive. Given $G = (V, E)$, we search for a specific vertex, called a *leaf-vertex*, whose closed neighborhood must be a leaf-bag of a star-decomposition if $tb(G) = 1$. Basing on Lemma 5 and a delicate case-by-case analysis of the structure of star-decompositions, we define three types of leaf-vertices (*e.g.*, see Figure 3). A vertex v is a *leaf-vertex* if one of the following conditions hold.

Type 1. $N(v)$ induces an $a_v b_v$-path for some $a_v, b_v \in V \setminus \{v\}$, denoted by Π_v, of length at least 3 and there is $d_v \in V \setminus \{v\}$ such that $N(v) \subseteq N(d_v)$.
Type 2. $N(v)$ induces a path, denoted by $\Pi_v = (a_v, b_v, c_v)$, of length 2.
Type 3. $N(v)$ consists of two non adjacent vertices a_v and c_v, and there is $b_v \in (N(a_v) \cap N(c_v)) \setminus \{v\}$.

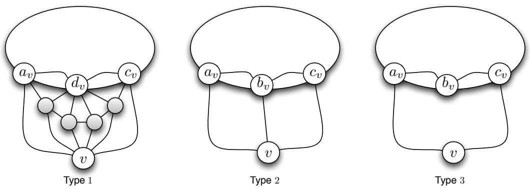


Fig. 3. The three kinds of leaf-vertices.

Ideally, we would like to remove v from G and apply recursively our algorithm on $G \setminus v$. However, in some case $tb(G \setminus v) = 1$ while $tb(G) > 1$ (see Fig. 2). So, we must also add edges between vertices that must be in a common bag of a star-decomposition of G if $tb(G) = 1$[4]. The choice of the edges to add is made

[4] We aim at turning the separator $N(v)$ into a clique. However, we cannot do that directly since it would break the distances in G, and the graph needs to stay planar.

more difficult by the need for the resulting graph G' to stay prime and planar in order to apply our algorithm recursively on G'. To show that $tb(G) = 1$ if and only if the resulting graph has treebreadth one also requires tedious lemmas.

Theorem 4. *Recognizing planar graphs of treebreadth one can be done in quadratic time. Moreover, a star-decomposition (if any) can be computed in cubic time.*

Sketch proof. Let $G = (V, E)$ be a prime planar graph. We can assume $|V| \geq 8$ and G has no star-decomposition with two bags (both cases are treated separately by exhaustive search). In such case, $tb(G) = 1$ implies there exists a leaf-vertex v, that can be found in linear time.

If $G \setminus v$ is prime then we prove $tb(G) = 1$ if and only if $tb(G \setminus v) = 1$, except in the special case when v is of Type 2 or 3 and $|(N(a_v) \cap N(c_v)) \setminus v| \leq 2$. Furthermore, we prove for the latter case that a_v, c_v must have two common neighbours u_v, b_v in $G \setminus v$ (else, $tb(G) > 1$) and G', obtained from G by adding the edges $\{v, u_v\}, \{v, b_v\}$, is planar and prime, and it satisfies $tb(G) = 1$ if and only if $tb(G') = 1$. So, we call the algorithm either on G' or on $G \setminus v$[5].

The most difficult situation is when $G \setminus v$ contains a clique-separator. This case is reduced to the one when v is of Type 2, there is an edge-separator (b_v, u_v) of $G \setminus v$, and $\{a_v, u_v\} \notin E$. Then, we aim at applying the algorithm recursively on G', obtained from $G \setminus v$ by adding the edge $\{a_v, c_v\}$. However, $tb(G') = 1$ does not imply $tb(G) = 1$ in general. We prove it is the case if u_v, c_v are nonadjacent or $N(u_v) \cap N(a_v)$ does not disconnect a_v from u_v in $G \setminus (c_v, v)$.

Else, we compute a plane embedding of G, and a vertex $x \in N(a_v) \cap N(u_v)$ such that: v, c_v and all other common neighbours of a_v, u_v are in a same region $\mathcal{R}$, bounded by (a_v, x, u_v, b_v). We wish to create an $a_v u_v$-path in $V \setminus \mathcal{R}$ by adding edges in $N(b_v) \cap N(x)$. In doing so, we go back to the previous subcase as now $N(a_v) \cap N(u_v)$ is no more a $a_v u_v$-separator of $G \setminus (c_v, v)$. However, we have to ensure that it is possible to add such a path in $V \setminus \mathcal{R}$, and that its addition does not affect the value of treebreadth for the graph. We prove it is the case unless $V \subseteq \mathcal{R}$ (in which case we apply the algorithm recursively on G', obtained from G by identifying b_v with x), or if there is a leaf-vertex $l \in N(b_v) \cap N(x)$. Furthermore, in the latter case we replace v with l in the above analysis, *i.e.*, l becomes the actual leaf-vertex to be considered.

Additional properties are needed in order to prove the algorithm terminates, and that it does so in a linear number of steps. □

Conclusion. We conclude this paper by some questions that remain open. First, it would be interesting to know the complexity of deciding the treebreadth of planar graphs. Second, all the reductions presented in this paper rely on constructions containing large clique or clique-minor. We left open the problem of recognizing graphs with tree-breadth one in the class of graphs with bounded treewidth or bounded clique-number. More generally, is the problem of computing the treebreadth Fixed-Parameter Tractable when it is parameterized by the treewidth or by the size of a largest clique-minor?

[5] When v is of Type 1 we call the algorithm on G', obtained from $G \setminus v$ by contracting the internal nodes of Π_v to an edge, in order to obtain a quadratic complexity.

References

1. M. Abu-Ata and F. Dragan. Metric tree-like structures in real-world networks: an empirical study. *Networks*, 67:49–68, 2016.
2. S. Arnborg, D. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic Discrete Methods*, 8(2):277–284, 1987.
3. A. Berry, R. Pogorelcnik, and G. Simonet. An introduction to clique minimal separator decomposition. *Algorithms*, 3(2):197–215, 2010.
4. H. Bodlaender. Treewidth: Characterizations, applications, and computations.
5. H. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.
6. H. Bodlaender, M. Fellows, and T. Warnow. Two strikes against perfect phylogeny. In *ICALP'92, Vienna, Austria*, pages 273–283.
7. A. Brandstädt, F. Dragan, V. Chepoi, and V. Voloshin. Dually chordal graphs. *SIAM Journal on Discrete Mathematics*, 11(3):437–455, 1998.
8. S. Chechik, D. Larkin, L. Roditty, G. Schoenebeck, R. Tarjan, and V. Vassilevska Williams. Better approximation algorithms for the graph diameter. In *ACM SODA'14*, pages 1041–1052. SIAM.
9. V. Chepoi, F. Dragan, B. Estellon, M. Habib, and Y. Vaxès. Diameters, centers, and approximating trees of δ-hyperbolic geodesic spaces and graphs. In *SCG '08*, pages 59–68, New York, NY, USA. ACM.
10. D. Coudert, G. Ducoffe, and N. Nisse. Diameter of Minimal Separators in Graphs. Research Report RR-8639, Inria Sophia Antipolis ; I3S ; INRIA, Nov. 2014.
11. F. de Montgolfier, M. Soto, and L. Viennot. Treewidth and hyperbolicity of the internet. In *Network Computing and Applications (NCA), 2011 10th IEEE International Symposium on*, pages 25–32, Aug 2011.
12. Y. Dourisboure, F. Dragan, C. Gavoille, and Y. Chenyu. Spanners for bounded tree-length graphs. *Theor. Comput. Sci.*, 383(1):34–44, 2007.
13. Y. Dourisboure and C. Gavoille. Tree-decompositions with bags of small diameter. *Discrete Mathematics*, 307(16):2008–2029, 2007.
14. F. Dragan and E. Köhler. An approximation algorithm for the tree t-spanner problem on unweighted graphs via generalized chordal graphs. *Algorithmica*, 69(4):884–905, 2014.
15. F. Dragan, E. Köhler, and A. Leitert. Line-distortion, bandwidth and path-length of a graph. In *Algorithm Theory–SWAT 2014*, pages 158–169. Springer, 2014.
16. F. Dragan and A. Leitert. On the minimum eccentricity shortest path problem. In *Algorithms and Data Structures – WADS*, pages 276–288. Springer, 2015.
17. G. Ducoffe, S. Legay, and N. Nisse. On computing tree and path decompositions with metric constraints on the bags. Technical Report RR-8842, 2016.
18. R. Krauthgamer and J. Lee. Algorithms on negatively curved spaces. In *FOCS'06*, pages 119–132. IEEE, 2006.
19. D. Lokshtanov. On the complexity of computing treelength. *Discrete Applied Mathematics*, 158(7):820–827, 2010.
20. N. Robertson and P. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of algorithms*, 7(3):309–322, 1986.
21. C. Wang, T. Liu, W. Jiang, and K. Xu. Feedback vertex sets on tree convex bipartite graphs. In *COCOA 2012, Banff, AB, Canada*, pages 95–102, 2012.
22. Y. Wu, P. Austrin, T. Pitassi, and D. Liu. Inapproximability of treewidth and related problems. *J. Artif. Intell. Res. (JAIR)*, 49:569–600, 2014.