Theoretical Computer Science 547 (2014) 1–17

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Collective additive tree spanners of bounded tree-breadth graphs with generalizations and consequences ☆

Feodor F. Dragan *, Muad Abu-Ata

Algorithmic Research Laboratory, Department of Computer Science, Kent State University, Kent, OH 44242, USA

ARTICLE INFO

Article history:
Received 12 October 2012
Received in revised form 5 December 2013
Accepted 4 June 2014
Available online 12 June 2014
Communicated by P. Widmayer

Keywords:
Graph algorithms
Approximation algorithms
Tree spanner problem
Collective tree spanners
Spanners of bounded tree-width
Robertson–Seymour's tree-decomposition
Balanced separators

ABSTRACT

In this paper, we study collective additive tree spanners for families of graphs enjoying special Robertson–Seymour's tree-decompositions, and demonstrate interesting consequences of obtained results. We say that a graph G *admits a system of* μ *collective additive tree* r*-spanners* (resp., *multiplicative tree* t*-spanners*) if there is a system $\mathcal{T}(G)$ of at most μ spanning trees of G such that for any two vertices x, y of G a spanning tree $T \in \mathcal{T}(G)$ exists such that $d_T(x, y) \le d_G(x, y) + r$ (resp., $d_T(x, y) \le t \cdot d_G(x, y)$). When $\mu = 1$ one gets the notion of *additive tree* r*-spanner* (resp., *multiplicative tree* t*-spanner*). It is known that if a graph G has a multiplicative tree t-spanner, then G admits a Robertson–Seymour's tree-decomposition with bags of radius at most $\lceil t/2 \rceil$ in G. We use this to demonstrate that there is a polynomial time algorithm that, given an n-vertex graph G admitting a multiplicative tree t-spanner, constructs a system of at most $\log_2 n$ collective additive tree $O(t \log n)$-spanners of G. That is, with a slight increase in the number of trees and in the stretch, one can "turn" a multiplicative tree spanner into a small set of collective additive tree spanners. We extend this result by showing that if a graph G admits a multiplicative t-spanner with tree-width $k-1$, then G admits a Robertson–Seymour's tree-decomposition each bag of which can be covered with at most k disks of G of radius at most $\lceil t/2 \rceil$ each. This is used to demonstrate that, for every fixed k, there is a polynomial time algorithm that, given an n-vertex graph G admitting a multiplicative t-spanner with tree-width $k-1$, constructs a system of at most $k(1 + \log_2 n)$ collective additive tree $O(t \log n)$-spanners of G.

1. Introduction

One of the basic questions in the design of routing schemes for communication networks is to construct a spanning network (a so-called spanner) which has two (often conflicting) properties: it should have simple structure and nicely approximate distances in the network. This problem fits in a larger framework of combinatorial and algorithmic problems that are concerned with distances in a finite metric space induced by a graph. An arbitrary metric space (in particular a finite metric defined by a graph) might not have enough structure to exploit algorithmically. A powerful technique that has been successfully used recently in this context is to embed the given metric space in a simpler metric space such that the distances are approximately preserved in the embedding. New and improved algorithms have resulted from this idea for several important problems (see, e.g., [4,6,18,39,45,52]).

☆ Results of this paper were partially presented at the SOFSEM 2013 conference [27].

* Corresponding author.
E-mail addresses: dragan@cs.kent.edu (F.F. Dragan), mabuata@cs.kent.edu (M. Abu-Ata).

http://dx.doi.org/10.1016/j.tcs.2014.06.007

There are several ways to measure the quality of this approximation, two of them leading to the notion of a spanner. For $t \geq 1$, a spanning subgraph H of $G = (V, E)$ is called a *(multiplicative) t-spanner* of G if $d_H(u, v) \leq t \cdot d_G(u, v)$ for all $u, v \in V$ [19,56,57]. If $r \geq 0$ and $d_H(u, v) \leq d_G(u, v) + r$, for all $u, v \in V$, then H is called an *additive r-spanner* of G [51,60,61]. The parameter t is called the *stretch* (or *stretch factor*) of H, while the parameter r is called the *surplus* of H. In what follows, we will often omit the word "multiplicative" when we refer to multiplicative spanners.

Tree metrics are a very natural class of simple metric spaces since many algorithmic problems become tractable on them. A *(multiplicative) tree t-spanner* of a graph G is a spanning tree with a stretch t [17], and an *additive tree r-spanner* of G is a spanning tree with a surplus r [60]. If we approximate the graph by a tree spanner, we can solve a given problem on the tree and the solution interpret on the original graph. The TREE t-SPANNER problem asks, given a graph G and a positive number t, whether G admits a tree t-spanner. Note that the problem of finding a tree t-spanner of G minimizing t is known in the literature also as the Minimum Max-Stretch spanning Tree problem (see, e.g., [40] and literature cited therein).

Unfortunately, not many graph families admit good tree spanners. This motivates the study of sparse spanners, i.e., spanners with a small amount of edges. There are many applications of spanners in various areas; especially, in distributed systems and communication networks. In [58], close relationships were established between the quality of spanners (in terms of stretch factor and the number of spanner edges), and the time and communication complexities of any synchronizer for the network based on this spanner. Another example is the usage of tree t-spanners in the analysis of arrow distributed queuing protocols [47,55]. Sparse spanners are very useful in message routing in communication networks; in order to maintain succinct routing tables, efficient routing schemes can use only the edges of a sparse spanner [59]. The SPARSEST t-SPANNER problem asks, for a given graph G and a number t, to find a t-spanner of G with the smallest number of edges. We refer to the survey paper of Peleg [54] for an overview on spanners.

Inspired by ideas from works of Alon et al. [1], Bartal [4,5], Fakcharoenphol et al. [41], and to extend those ideas to designing compact and efficient routing and distance labeling schemes in networks, in [31], a new notion of *collective tree spanners*[1] was introduced. This notion is slightly *weaker* than the one of a tree spanner and slightly *stronger* than the notion of a sparse spanner. We say that a graph $G = (V, E)$ *admits a system of μ collective additive tree r-spanners* if there is a system $\mathcal{T}(G)$ of at most μ spanning trees of G such that for any two vertices x, y of G a spanning tree $T \in \mathcal{T}(G)$ exists such that $d_T(x, y) \leq d_G(x, y) + r$ (a multiplicative variant of this notion can be defined analogously). Clearly, if G admits a system of μ collective additive tree r-spanners, then G admits an additive r-spanner with at most $\mu \times (n - 1)$ edges (take the union of all those trees), and if $\mu = 1$ then G admits an additive tree r-spanner.

Recently, in [32], *spanners of bounded tree-width* were introduced, motivated by the fact that many algorithmic problems are tractable on graphs of bounded tree-width, and a spanner H of G with small tree-width can be used to obtain an approximate solution to a problem on G. In particular, efficient and compact distance and routing labeling schemes are available for bounded tree-width graphs (see, e.g., [29,45] and papers cited therein), and they can be used to compute approximate distances and route along paths that are close to shortest in G. The k-TREE-WIDTH t-SPANNER problem asks, for a given graph G, an integer k and a positive number $t \geq 1$, whether G admits a t-spanner of tree-width at most k. Every connected graph with n vertices and at most $n - 1 + m$ edges is of tree-width at most $m + 1$ and hence this problem is a generalization of the TREE t-SPANNER and the SPARSEST t-SPANNER problems. Furthermore, t-spanners of bounded tree-width have much more structure to exploit algorithmically than sparse t-spanners (which have a small number of edges but may lack other nice structural properties).

1.1. Related work

Tree spanners. Substantial work has been done on the TREE t-SPANNER problem on unweighted graphs. Cai and Corneil [17] have shown that, for a given graph G, the problem to decide whether G has a tree t-spanner is NP-complete for any fixed $t \geq 4$ and is linear time solvable for $t = 1, 2$ (the status of the case $t = 3$ is open for general graphs).[2] The NP-completeness result was further strengthened in [15] and [16], where Brandstädt et al. showed that the problem remains NP-complete even for the class of chordal graphs (i.e., for graphs where each induced cycle has length 3) and every fixed $t \geq 4$, and for the class of chordal bipartite graphs (i.e., for bipartite graphs where each induced cycle has length 4) and every fixed $t \geq 5$.

The TREE t-SPANNER problem on planar graphs was studied in [32,42]. In [42], Fekete and Kremer proved that the TREE t-SPANNER problem on planar graphs is NP-complete (when t is part of the input) and polynomial time solvable for $t = 3$. For fixed $t \geq 4$, the complexity of the TREE t-SPANNER problem on arbitrary planar graphs was left as an open problem in [42]. This open problem was recently resolved in [32] by Dragan et al., where it was shown that the TREE t-SPANNER problem is linear time solvable for every fixed constant t on the class of apex-minor-free graphs which includes all planar graphs and all graphs of bounded genus. Note also that a number of particular graph classes (like interval graphs, permutation graphs, asteroidal-triple-free graphs, strongly chordal graphs, dually chordal graphs, and others) admit additive tree r-spanners for small values of r (we refer reader to [14–17,42,50,54,55,60,61] and papers cited therein).

The first $O(\log n)$-approximation algorithm for the minimum value of t for the TREE t-SPANNER problem was developed by Emek and Peleg in [40] (where n is the number of vertices in a graph). Recently, another logarithmic approximation

[1] Independently, Gupta et al. in [45] introduced a similar concept which is called *tree covers* there.
[2] When G is an unweighted graph, t can be assumed to be an integer.

algorithm for the problem was proposed in [28] (we elaborate more on this in Section 1.2). Emek and Peleg also established in [40] that unless P = NP, the problem cannot be approximated additively by any $o(n)$ term. Hardness of approximation is established also in [50], where it was shown that approximating the minimum value of t for the TREE t-SPANNER problem within factor better than 2 is NP-hard (see also [55] for an earlier result).

Sparse spanners. Sparse t-spanners were introduced by Peleg, Schäffer and Ullman in [56,58] and since that time were studied extensively. It was shown by Peleg and Schäffer in [56] that the problem of deciding whether a graph G has a t-spanner with at most m edges is NP-complete. Later, Kortsarz [48] showed that for every $t \geq 2$ there is a constant $c < 1$ such that it is NP-hard to approximate the sparsest t-spanner within the ratio $c \cdot \log n$, where n is the number of vertices in the graph. On the other hand, the problem admits an $O(\log n)$-ratio approximation for $t = 2$ [49,48] and an $O(n^{2/(t+1)})$-ratio approximation for $t > 2$ [37]. For some other inapproximability and approximability results for the SPARSEST t-SPANNER problem on general graphs we refer the reader to [11,12,23,22,38,35,37,63] and papers cited therein. It is interesting to note also that any (even weighted) n-vertex graph admits an $O(2k-1)$-spanner with at most $O(n^{1+1/k})$ edges for any $k \geq 1$, and such a spanner can be constructed in polynomial time [2,7,63].

On planar graphs the SPARSEST t-SPANNER problem was studied as well. Brandes and Handke have shown that the decision version of the problem remains NP-complete on planar graphs for every fixed $t \geq 5$ (the case $2 \leq t \leq 4$ is open) [13]. Duckworth, Wormald, and Zito [34] have shown that the problem of finding a sparsest 2-spanner of a 4-connected planar triangulation admits a polynomial time approximation scheme (PTAS). Dragan et al. [33] proved that the SPARSEST t-SPANNER problem admits PTAS for graph classes of bounded local tree-width (and therefore for planar and bounded genus graphs).

Sparse additive spanners were considered in [8,24,36,51,64]. It is known that every n-vertex graph admits an additive 2-spanner with at most $\Theta(n^{3/2})$ edges [24,36], an additive 6-spanner with at most $O(n^{4/3})$ edges [8], and an additive $O(n^{(1-1/k)/2})$-spanner with at most $O(n^{1+1/k})$ edges for any $k \geq 1$ [8]. All those spanners can be constructed in polynomial time. We refer the reader to paper [64] for a good summary of the state of the art of results on the sparsest additive spanner problem in general graphs.

Collective tree spanners. The problem of finding "small" systems of collective additive tree r-spanners for small values of r was examined on special classes of graphs in [20,30,29,31,65]. For example, in [20,31], sharp results were obtained for unweighted chordal graphs and c-chordal graphs (i.e., the graphs where each induced cycle has length at most c): every c-chordal graph admits a system of at most $\log_2 n$ collective additive tree $(2\lfloor c/2 \rfloor)$-spanners, constructible in polynomial time; no system of constant number of collective additive tree r-spanners can exist for chordal graphs (i.e., when $c = 3$) and $r \leq 3$, and no system of constant number of collective additive tree r-spanners can exist for outerplanar graphs for any constant r.

Only papers [29,45,65] have investigated collective (multiplicative or additive) tree spanners in *weighted graphs*. It was shown that any weighted n-vertex planar graph admits a system of $O(\sqrt{n})$ collective multiplicative tree 1-spanners (equivalently, additive tree 0-spanners) [29,45] and a system of at most $2\log_{3/2} n$ collective multiplicative tree 3-spanners [45]. Furthermore, any weighted graph with genus at most g admits a system of $O(\sqrt{gn})$ collective additive tree 0-spanners [29,45], any weighted graph with tree-width at most $k-1$ admits a system of at most $k \log_2 n$ collective additive tree 0-spanners [29,45], any weighted graph G with clique-width at most k admits a system of at most $k \log_{3/2} n$ collective additive tree (2w)-spanners [29], any weighted c-chordal graph G admits a system of $\log_2 n$ collective additive tree $(2\lfloor c/2 \rfloor \mathrm{w})$-spanners [29] (where w denotes the maximum edge weight in G).

Collective tree spanners of Unit Disk Graphs (UDGs) (which often model wireless ad hoc networks) were investigated in [65]. It was shown that every n-vertex UDG G admits a system $\mathcal{T}(G)$ of at most $2\log_{\frac{3}{2}} n + 2$ spanning trees of G such that, for any two vertices x and y of G, there exists a tree T in $\mathcal{T}(G)$ with $d_T(x, y) \leq 3 \cdot d_G(x, y) + 12$. That is, the distances in any UDG can be approximately represented by the distances in at most $2\log_{\frac{3}{2}} n + 2$ of its spanning trees. Based on this result a new *compact and low delay routing labeling scheme* was proposed for Unit Disk Graphs.

Spanners with bounded tree-width. The k-TREE-WIDTH t-SPANNER problem was considered in [32] and [43]. It was shown that the problem is linear time solvable for every fixed constants t and k on the class of apex-minor-free graphs [32], which includes all planar graphs and all graphs of bounded genus, and on the graphs with bounded degree [43].

1.2. Our results and their place in the context of the previous results

This paper was inspired by few recent results from [26,28,37,40]. Elkin and Peleg in [37], among other results, described a polynomial time algorithm that, given an n-vertex graph G admitting a tree t-spanner, constructs a t-spanner of G with $O(n \log n)$ edges. Emek and Peleg in [40] presented the first $O(\log n)$-approximation algorithm for the minimum value of t for the TREE t-SPANNER problem. They described a polynomial time algorithm that, given an n-vertex graph G admitting a tree t-spanner, constructs a tree $O(t \log n)$-spanner of G. Later, a simpler and faster $O(\log n)$-approximation algorithm for the problem was given by Dragan and Köhler [28]. Their result uses a new necessary condition for a graph to have a tree t-spanner: if a graph G has a tree t-spanner, then G admits a Robertson–Seymour's tree-decomposition with bags of radius at most $\lceil t/2 \rceil$ in G.

To describe the results of [26] and to elaborate more on the Dragan–Köhler's approach, we need to recall definitions of a few graph parameters. They all are based on the notion of tree-decomposition introduced by Robertson and Seymour in their work on graph minors [62].

A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(\{X_i | i \in I\}, T = (I, F))$ where $\{X_i | i \in I\}$ is a collection of subsets of V, called *bags*, and T is a tree. The nodes of T are the bags $\{X_i | i \in I\}$ satisfying the following three conditions:

1. $\bigcup_{i \in I} X_i = V$;
2. for each edge $uv \in E$, there is a bag X_i such that $u, v \in X_i$;
3. for all $i, j, k \in I$, if j is on the path from i to k in T, then $X_i \cap X_k \subseteq X_j$. Equivalently, this condition could be stated as follows: for all vertices $v \in V$, the set of bags $\{i \in I | v \in X_i\}$ induces a connected subtree T_v of T.

For simplicity we denote a tree-decomposition $(\{X_i | i \in I\}, T = (I, F))$ of a graph G by $T(G)$.

Tree-decompositions were used to define several graph parameters to measure how close a given graph is to some known graph class (e.g., to trees or to chordal graphs) where many algorithmic problems could be solved efficiently. The *width* of a tree-decomposition $T(G) = (\{X_i | i \in I\}, T = (I, F))$ is $\max_{i \in I} |X_i| - 1$. The *tree-width* of a graph G, denoted by $\mathrm{tw}(G)$, is the minimum width, over all tree-decompositions $T(G)$ of G [62]. The trees are exactly the graphs with tree-width 1. The *length* of a tree-decomposition $T(G)$ of a graph G is $\lambda := \max_{i \in I} \max_{u,v \in X_i} d_G(u, v)$ (i.e., each bag X_i has diameter at most λ in G). The *tree-length* of G, denoted by $\mathrm{tl}(G)$, is the minimum of the length, over all tree-decompositions of G [25]. The chordal graphs are exactly the graphs with tree-length 1. Note that these two graph parameters are not related to each other. For instance, a clique on n vertices has tree-length 1 and tree-width $n - 1$, whereas a cycle on $3n$ vertices has tree-width 2 and tree-length n. In [28], yet another graph parameter was introduced, which is very similar to the notion of tree-length and, as it turns out, is related to the TREE t-SPANNER problem. The *breadth* of a tree-decomposition $T(G)$ of a graph G is the minimum integer r such that for every $i \in I$ there is a vertex $v_i \in V(G)$ with $X_i \subseteq D_r(v_i, G)$ (i.e., each bag X_i can be covered by a disk $D_r(v_i, G) := \{u \in V(G) | d_G(u, v_i) \le r\}$ of radius at most r in G). Note that vertex v_i does not need to belong to X_i. The *tree-breadth* of G, denoted by $\mathrm{tb}(G)$, is the minimum of the breadth, over all tree-decompositions of G. Evidently, for any graph G, $1 \le \mathrm{tb}(G) \le \mathrm{tl}(G) \le 2\mathrm{tb}(G)$ holds. Hence, if one parameter is bounded by a constant for a graph G then the other parameter is bounded for G as well.

We say that a family of graphs $\mathcal{G}$ is *of bounded tree-breadth* (*of bounded tree-width, of bounded tree-length*) if there is a constant c such that for each graph G from $\mathcal{G}$, $\mathrm{tb}(G) \le c$ (resp., $\mathrm{tw}(G) \le c$, $\mathrm{tl}(G) \le c$).

It was shown in [28] that if a graph G admits a tree t-spanner then its tree-breadth is at most $\lceil t/2 \rceil$ and its tree-length is at most t. Furthermore, any graph G with tree-breadth $\mathrm{tb}(G) \le \rho$ admits a tree $(2\rho \lfloor \log_2 n \rfloor)$-spanner that can be constructed in polynomial time. Thus, these two results gave a new $\log_2 n$-approximation algorithm for the TREE t-SPANNER problem on general (unweighted) graphs (see [28] for details). The algorithm of [28] is conceptually simpler than the previous $O(\log n)$-approximation algorithm proposed for the problem by Emek and Peleg [40].

Dourisboure et al. in [26] considered the construction of additive spanners with few edges for n-vertex graphs having a tree-decomposition into bags of diameter at most λ, i.e., the tree-length λ graphs. For such graphs they construct additive 2λ-spanners with $O(\lambda n + n \log n)$ edges, and additive 4λ-spanners with $O(\lambda n)$ edges. Combining these results with the results of [28], we obtain the following interesting fact (in a sense, turning a multiplicative stretch into an additive surplus without much increase in the number of edges).

Theorem 1. *(Combining [26] and [28].) If a graph G admits a (multiplicative) tree t-spanner then it has an additive $2t$-spanner with $O(tn + n \log n)$ edges and an additive $4t$-spanner with $O(tn)$ edges, both constructible in polynomial time.*

This fact rises few intriguing questions. Does a polynomial time algorithm exist that, given an n-vertex graph G admitting a (multiplicative) tree t-spanner, constructs an additive $O(t)$-spanner of G with $O(n)$ or $O(n \log n)$ edges (where the number of edges in the spanner is independent of t)? Is a result similar to the one presented by Elkin and Peleg in [37] possible? Namely, does a polynomial time algorithm exist that, given an n-vertex graph G admitting a (multiplicative) tree t-spanner, constructs an additive $(t - 1)$-spanner[3] of G with $O(n \log n)$ edges? If we allow to use more trees (like in collective tree spanners), does a polynomial time algorithm exist that, given an n-vertex graph G admitting a (multiplicative) tree t-spanner, constructs a system of $\tilde{O}(1)$ collective additive tree $\tilde{O}(t)$-spanners of G (where $\tilde{O}$ is similar to Big-O notation up to a poly-logarithmic factor)? Note that an interesting question whether a multiplicative tree spanner can be turned into an additive tree spanner with a slight increase in the stretch is (negatively) settled already in [40]: if there exist some $\delta = o(n)$ and $\epsilon > 0$ and a polynomial time algorithm that for any graph admitting a tree t-spanner constructs a tree $((6/5 - \epsilon)t + \delta)$-spanner, then $P = NP$.

We give some partial answers to these questions in Section 3. There we investigate a more general question whether a graph with bounded tree-breadth admits a small system of collective additive tree spanners. We show that any n-vertex graph G has a system of at most $\log_2 n$ collective additive tree $(2\rho \log_2 n)$-spanners, where $\rho \le \mathrm{tb}(G)$. This settles also an open question from [26] whether a graph with tree-length λ admits a small system of collective additive tree $\tilde{O}(\lambda)$-spanners.

[3] Note that any additive $(t - 1)$-spanner is a multiplicative t-spanner (see Proposition 2).

As a consequence, we obtain that there is a polynomial time algorithm that, given an n-vertex graph G admitting a (multiplicative) tree t-spanner, constructs:

- a system of at most $\log_2 n$ collective additive tree $O(t \log n)$-spanners of G (compare with [28,40] where a multiplicative tree $O(t \log n)$-spanner was constructed for G in polynomial time; thus, we "have turned" a multiplicative tree $O(t \log n)$-spanner into at most $\log_2 n$ collective additive tree $O(t \log n)$-spanners);
- an additive $O(t \log n)$-spanner of G with at most $n \log_2 n$ edges (compare with Theorem 1).

In Section 4 we generalize the method of Section 3. We define a new notion which combines both the tree-width and the tree-breadth of a graph.

The *k-breadth* of a tree-decomposition $T(G) = (\{X_i | i \in I\}, T = (I, F))$ of a graph G is the minimum integer r such that for each bag $X_i, i \in I$, there is a set of at most k vertices $C_i = \{v^i_j | v^i_j \in V(G), j = 1, \ldots, k\}$ such that for each $u \in X_i$, we have $d_G(u, C_i) \le r$ (i.e., each bag X_i can be covered with at most k disks of G of radius at most r each; $X_i \subseteq D_r(v^i_1, G) \cup \cdots \cup D_r(v^i_k, G)$). The *k-tree-breadth* of a graph G, denoted by $\mathrm{tb}_k(G)$, is the minimum of the k-breadth, over all tree-decompositions of G. We say that a family of graphs $\mathcal{G}$ is *of bounded k-tree-breadth*, if there is a constant c such that for each graph G from $\mathcal{G}$, $\mathrm{tb}_k(G) \le c$. Clearly, for every graph G, $\mathrm{tb}(G) = \mathrm{tb}_1(G)$, and $\mathrm{tw}(G) \le k-1$ if and only if $\mathrm{tb}_k(G) = 0$ (each vertex in the bags of the tree-decomposition of width k can be considered as a center of a disk of radius 0). Thus, the notions tree-width and the tree-breadth are particular cases of the k-tree-breadth.

In Section 4, we show that any n-vertex graph G with $\mathrm{tb}_k(G) \le \rho$ has a system of at most $k(1 + \log_2 n)$ collective additive tree $(2\rho(1 + \log_2 n))$-spanners. In Section 5, we extend a result from [28] and show that if a graph G admits a (multiplicative) t-spanner H with $\mathrm{tw}(H) = k-1$ then its k-tree-breadth is at most $\lceil t/2 \rceil$. As a consequence, we obtain that, for every fixed k, there is a polynomial time algorithm that, given an n-vertex graph G admitting a (multiplicative) t-spanner with tree-width at most $k-1$, constructs:

- a system of at most $k(1 + \log_2 n)$ collective additive tree $O(t \log n)$-spanners of G;
- an additive $O(t \log n)$-spanner of G with at most $O(kn \log n)$ edges.

We conclude the paper with some open questions.

2. Preliminaries

All graphs occurring in this paper are connected, finite, unweighted, undirected, loopless and without multiple edges. We call $G = (V, E)$ an *n-vertex m-edge graph* if $|V| = n$ and $|E| = m$. A *clique* is a set of pairwise adjacent vertices of G. By $G[S]$ we denote a subgraph of G induced by vertices of $S \subseteq V$. Let also $G \setminus S$ be the graph $G[V \setminus S]$ (which is not necessarily connected). A set $S \subseteq V$ is called a *separator* of a connected graph G if the graph $G[V \setminus S]$ has more than one connected component, and S is called a *balanced separator* of G if each connected component of $G[V \setminus S]$ has at most $|V|/2$ vertices. A set $C \subseteq V$ is called a *balanced clique-separator* of G if C is both a clique and a balanced separator of G. For a vertex v of G, the sets $N_G(v) = \{w \in V | vw \in E\}$ and $N_G[v] = N_G(v) \cup \{v\}$ are called the *open neighborhood* and the *closed neighborhood* of v, respectively.

In a graph G the *length* of a path from a vertex v to a vertex u is the number of edges in the path. The *distance* $d_G(u, v)$ between vertices u and v is the length of a shortest path connecting u and v in G. The *diameter* in G of a set $S \subseteq V$ is $\max_{x,y \in S} d_G(x, y)$ and its *radius* in G is $\min_{x \in V} \max_{y \in S} d_G(x, y)$ (in some papers they are called the *weak diameter* and the *weak radius* to indicate that the distances are measured in G not in $G[S]$). The *disk* of G of radius k centered at vertex v is the set of all vertices at distance at most k to v: $D_k(v, G) = \{w \in V | d_G(v, w) \le k\}$. A disk $D_k(v, G)$ is called a *balanced disk-separator* of G if the set $D_k(v, G)$ is a balanced separator of G.

It is well known that the t-spanners can equivalently be defined as follows.

Proposition 1. *(See [17].) Let G be a connected graph and t be a positive number. A spanning subgraph H of G is a t-spanner of G if and only if for every edge xy of G, $d_H(x, y) \le t$ holds.*

This proposition implies that the stretch of a spanning subgraph of a graph G is always obtained on a pair of vertices that form an edge in G. Consequently, throughout this paper, t can be considered as an integer which is greater than 1 (the case $t = 1$ is trivial since H must be G itself).

It is also known that every additive r-spanner of G is a (multiplicative) $(r+1)$-spanner of G.

Proposition 2. *(See [60].) Every additive r-spanner of G is a (multiplicative) $(r+1)$-spanner of G. The converse is generally not true.*

3. Collective additive tree spanners and the tree-breadth of a graph

In this section, we show that every n-vertex graph G has a system of at most $\log_2 n$ collective additive tree $(2\rho \log_2 n)$-spanners, where $\rho \le \mathrm{tb}(G)$. We also discuss consequences of this result. Our method is a generalization of tech-

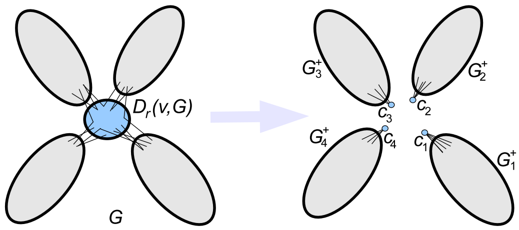


Fig. 1. A graph G with a disk-separator $D_r(v,G)$ and the corresponding graphs $G_1^+,\dots,G_4^+$ obtained from G. $c_1,\dots,c_4$ are meta vertices representing the disk $D_r(v,G)$ in the corresponding graphs.

niques used in [31] and [28]. We will assume that $n \ge 4$ since any connected graph with at most 3 vertices has an additive tree 1-spanner.

Note that we do not assume here that a tree-decomposition $T(G)$ of breadth ρ is given for G as part of the input. Our method does not need to know $T(G)$, our algorithm works directly on G. For a given graph G and an integer ρ, even checking whether G has a tree-decomposition of breadth ρ could be a hard problem. For example, while graphs with tree-length 1 (as they are exactly the chordal graphs) can be recognized in linear time, the problem of determining whether a given graph has tree-length at most λ is NP-complete for every fixed $\lambda > 1$ (see [53]).

We will need the following results proven in [28].

Lemma 1. *(See [28].) Every graph G has a balanced disk-separator $D_r(v,G)$ centered at some vertex v, where $r \le \mathrm{tb}(G)$.*

Lemma 2. *(See [28].) For an arbitrary graph G with n vertices and m edges a balanced disk-separator $D_r(v,G)$ with minimum r can be found in $O(nm)$ time.*

3.1. Hierarchical decomposition of a graph with bounded tree-breadth

In this subsection, following [28], we show how to decompose a graph with bounded tree-breadth and build a hierarchical decomposition tree for it. This hierarchical decomposition tree is used later for construction of collective additive tree spanners for such a graph.

Let $G=(V,E)$ be an arbitrary connected n-vertex m-edge graph with a disk-separator $D_r(v,G)$. Also, let $G_1,\dots,G_q$ be the connected components of $G[V \setminus D_r(v,G)]$. Denote by $S_i := \{x \in V(G_i) | d_G(x, D_r(v,G)) = 1\}$ the neighborhood of $D_r(v,G)$ with respect to G_i. Let also G_i^+ be the graph obtained from component G_i by adding a vertex c_i (*representative* of $D_r(v,G)$) and making it adjacent to all vertices of S_i, i.e., for a vertex $x \in V(G_i)$, $c_i x \in E(G_i^+)$ if and only if there is a vertex $x_D \in D_r(v,G)$ with $xx_D \in E(G)$. See Fig. 1 for an illustration. In what follows, we will call vertex c_i a *meta vertex representing disk $D_r(v,G)$ in graph G_i^+*. Given a graph G and its disk-separator $D_r(v,G)$, the graphs $G_1^+,\dots,G_q^+$ can be constructed in total time $O(m)$. Furthermore, the total number of edges in the graphs $G_1^+,\dots,G_q^+$ does not exceed the number of edges in G, and the total number of vertices (including q meta vertices) in those graphs does not exceed the number of vertices in $G[V \setminus D_r(v,G)]$ plus q.

Denote by $G_{/e}$ the graph obtained from G by contracting its edge e. Recall that edge e contraction is an operation which removes e from G while simultaneously merging together the two vertices e previously connected. If a contraction results in multiple edges, we delete duplicates of an edge to stay within the class of simple graphs. The operation may be performed on a set of edges by contracting each edge (in any order). The following lemma guarantees that the tree-breadths of the graphs G_i^+, $i=1,\dots,q$, are no larger than the tree-breadth of G.

Lemma 3. *(See [28].) For any graph G and its edge e, $\mathrm{tb}(G) \le \rho$ implies $\mathrm{tb}(G_{/e}) \le \rho$. Consequently, for any graph G with $\mathrm{tb}(G) \le \rho$, $\mathrm{tb}(G_i^+) \le \rho$ holds for each $i=1,\dots,q$.*

Clearly, one can get G_i^+ from G by repeatedly contracting (in any order) edges of G that are not incident to vertices of G_i. In other words, G_i^+ is a minor of G. Recall that a graph G' is a *minor* of G if G' can be obtained from G by contracting some edges, deleting some edges, and deleting some isolated vertices. The order in which a sequence of such contractions and deletions is performed on G does not affect the resulting graph G'.

Let $G=(V,E)$ be a connected n-vertex, m-edge graph and assume that $\mathrm{tb}(G) \le \rho$. Lemma 1 and Lemma 2 guarantee that G has a balanced disk-separator $D_r(v,G)$ with $r \le \rho$, which can be found in $O(nm)$ time by an algorithm that works directly on graph G and does not require construction of a tree-decomposition of G of breadth $\le \rho$. Using these and Lemma 3, we can build a (rooted) *hierarchical tree* $\mathcal{H}(G)$ for G as follows. If G is a connected graph with at most 5 vertices, then $\mathcal{H}(G)$ is one node tree with root node $(V(G),G)$. Otherwise, find a balanced disk-separator $D_r(v,G)$ in G with minimum r (see Lemma 2) and construct the corresponding graphs $G_1^+,G_2^+,\dots,G_q^+$. For each graph G_i^+ ($i=1,\dots,q$) (by Lemma 3,

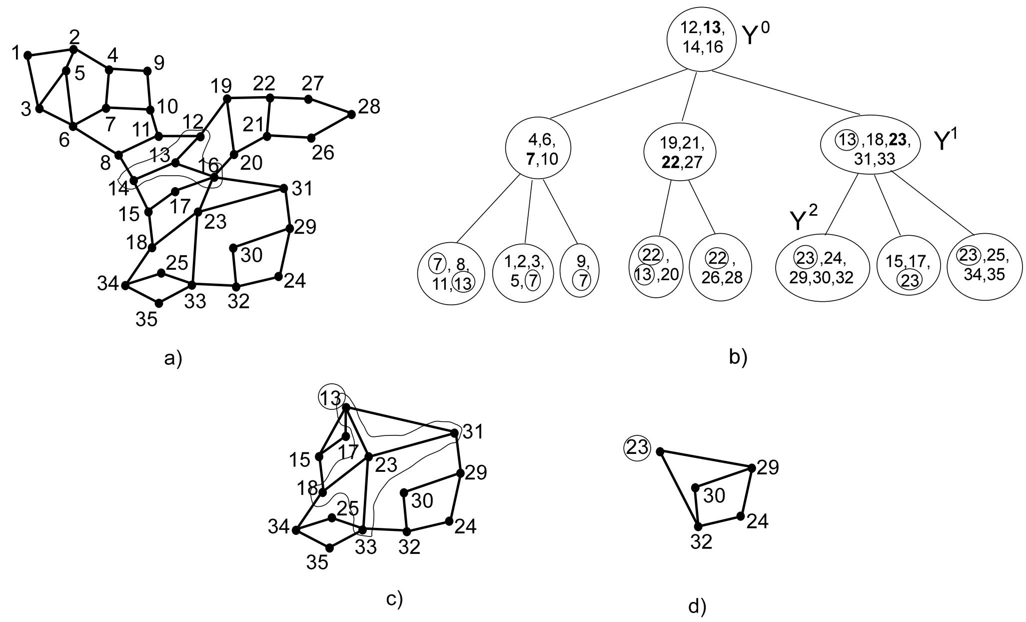


Fig. 2. a) A graph G and its balanced disk-separator $D_1(13, G)$. b) A hierarchical tree $\mathcal{H}(G)$ of G. We have $G = G(\downarrow Y^0)$, $Y^0 = D_1(13, G)$. Meta vertices are shown circled, disk centers are shown in bold. c) The graph $G(\downarrow Y^1)$ with its balanced disk-separator $D_1(23, G(\downarrow Y^1)) = Y^1$. $G(\downarrow Y^1)$ is a minor of $G(\downarrow Y^0)$. d) The graph $G(\downarrow Y^2)$, a minor of $G(\downarrow Y^1)$ and of $G(\downarrow Y^0)$. $Y^2 = V(G(\downarrow Y^2))$ is a leaf of $\mathcal{H}(G)$.

$\mathrm{tb}(G_i^+) \le \rho$), construct a hierarchical tree $\mathcal{H}(G_i^+)$ recursively and build $\mathcal{H}(G)$ by taking the pair $(D_r(v, G), G)$ to be the root and connecting the root of each tree $\mathcal{H}(G_i^+)$ as a child of $(D_r(v, G), G)$.

The depth of this tree $\mathcal{H}(G)$ (that is, the length of a longest path from the root to any node) is the smallest integer k such that

$$\frac{n}{2^k} + \frac{1}{2^{k-1}} + \cdots + \frac{1}{2} + 1 \le 5,$$

that is, the depth is at most $\log_2 n - 1$.

It is also easy to see that, given a graph G with n vertices and m edges, a hierarchical tree $\mathcal{H}(G)$ can be constructed in $O(nm\log^2 n)$ total time. There are at most $O(\log n)$ levels in $\mathcal{H}(G)$, and one needs to do at most $O(nm\log n)$ operations per level since the total number of edges in the graphs of each level is at most m and the total number of vertices in those graphs cannot exceed $O(n\log n)$.

For an internal (i.e., non-leaf) node Y of $\mathcal{H}(G)$, since it is associated with a pair $(D_{r'}(v', G'), G')$, where $r' \le \rho$, G' is a minor of G and v' is the center of disk $D_{r'}(v', G')$ of G', it will be convenient, in what follows, to denote G' by $G(\downarrow Y)$, v' by $c(Y)$, r' by $r(Y)$, and $D_{r'}(v', G')$ by Y itself. Thus, $(D_{r'}(v', G'), G') = (D_{r(Y)}(c(Y), G(\downarrow Y)), G(\downarrow Y)) = (Y, G(\downarrow Y))$ in these notations, and we identify node Y of $\mathcal{H}(G)$ with the set $Y = D_{r(Y)}(c(Y), G(\downarrow Y))$ and associate with this node also the graph $G(\downarrow Y)$. See Fig. 2 for an illustration. Each leaf Y of $\mathcal{H}(G)$, since it corresponds to a pair $(V(G'), G')$, we identify with the set $Y = V(G')$ and use, for convenience, the notation $G(\downarrow Y)$ for G'.

If now $(Y^0, Y^1, \ldots, Y^h)$ is the path of $\mathcal{H}(G)$ connecting the root Y^0 of $\mathcal{H}(G)$ with a node Y^h, then the vertex set of the graph $G(\downarrow Y^h)$ consists of some (original) vertices of G plus at most h meta vertices representing the disks $D_{r(Y)}(c(Y^i), G(\downarrow Y^i)) = Y^i$, $i = 0, 1, \ldots, h-1$. Note also that each (original) vertex of G belongs to exactly one node of $\mathcal{H}(G)$.

3.2. Construction of collective additive tree spanners

Unfortunately, the class of graphs of bounded tree-breadth is not hereditary, i.e., induced subgraphs of a graph with tree-breath ρ are not necessarily of tree-breadth at most ρ (for example, a cycle of length ℓ with one extra vertex adjacent to each vertex of the cycle has tree-breadth 1, but the cycle itself has tree-breadth $\ell/3$). Thus, the method presented in [31], for constructing collective additive tree spanners for hereditary classes of graphs admitting balanced disk-separators, cannot be applied directly to the graphs of bounded tree-breadth. Nevertheless, we will show that, with the help of Lemma 3, the notion hierarchical tree from the previous subsection and a careful analysis of distance changes (see Lemma 4), it is possible to generalize the method of [31] and construct in polynomial time for every n-vertex graph G a system of at most $\log_2 n$ collective additive tree $(2\rho\log_2 n)$-spanners, where $\rho \le \mathrm{tb}(G)$. Unavoidable presence of meta vertices in the graphs resulting from a hierarchical decomposition of the original graph G complicates the construction and the analysis. Recall that, in [31],

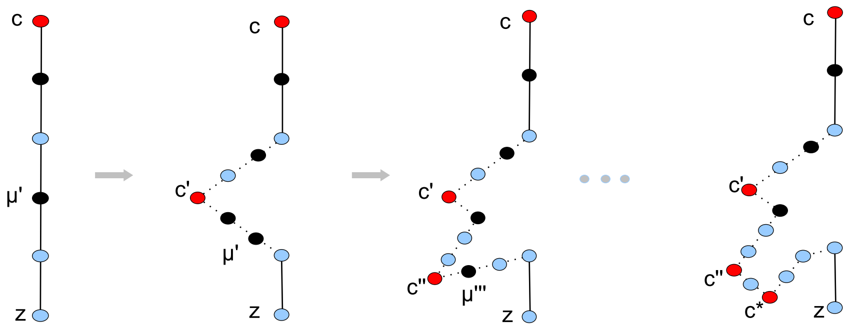


Fig. 3. Illustration to the proof of Lemma 4: "unfolding" meta vertices.

it was shown that if every induced subgraph of a graph G enjoys a balanced disk-separator with radius at most r, then G admits a system of at most $\log_2 n$ collective additive tree $2r$-spanners.

Let $G = (V, E)$ be a connected n-vertex, m-edge graph and assume that $\mathrm{tb}(G) \le \rho$. Let $\mathcal{H}(G)$ be a hierarchical tree of G. Consider an arbitrary internal node Y^h of $\mathcal{H}(G)$, and let $(Y^0, Y^1, \dots, Y^h)$ be the path of $\mathcal{H}(G)$ connecting the root Y^0 of $\mathcal{H}(G)$ with Y^h. Let $\widehat{G}(\downarrow Y^j)$ be the graph obtained from $G(\downarrow Y^j)$ by removing all its meta vertices (note that $\widehat{G}(\downarrow Y^j)$ may be disconnected).

Lemma 4. *For any vertex z from $Y^h \cap V(G)$ there exists an index $i \in \{0, 1, \dots, h\}$ such that $c(Y^i)$ is not a meta vertex and vertices z and $c(Y^i)$ are connected in the graph $\widehat{G}(\downarrow Y^i)$ by a path of length at most $\rho(h+1)$. In particular, $d_G(z, c(Y^i)) \le \rho(h+1)$ holds.*

Proof. Set $G_h := G(\downarrow Y^h)$, $c := c(Y^h)$, and let $SP^{G_h}_{c,z}$ be a shortest path of G_h connecting vertices c and z. We know that this path has at most $r(Y^h) \le \rho$ edges. If $SP^{G_h}_{c,z}$ does not contain any meta vertices, then this path is a path of $\widehat{G}(\downarrow Y^h)$ and of G and therefore $d_G(c, z) \le \rho$ holds.

Assume now that $SP^{G_h}_{c,z}$ does contain meta vertices and let μ' be the closest to z meta vertex in $SP^{G_h}_{c,z}$. See Fig. 3 for an illustration. Let $SP^{G_h}_{c,z} = (c, \dots, a', \mu', b', \dots, z)$. By construction of $\mathcal{H}(G)$, meta vertex μ' was created at some earlier recursive step to represent disk $Y^{i'}$ of graph $G_{i'} := G(\downarrow Y^{i'})$ for some $i' \in \{0, \dots, h-1\}$. Hence, there is a path $P^{G_{i'}}_{c',z} = (c', \dots, b', \dots, z)$ of length at most 2ρ in $G_{i'}$ with $c' := c(Y^{i'})$. Again, if $P^{G_{i'}}_{c',z}$ does not contain any meta vertices, then this path is a path of $\widehat{G}(\downarrow Y^{i'})$ and of G and therefore $d_G(c', z) \le 2\rho$ holds. If $P^{G_{i'}}_{c',z}$ does contain meta vertices then again, "unfolding" a meta vertex μ'' of $P^{G_{i'}}_{c',z}$ closest to z, we obtain a path $P^{G_{i''}}_{c'',z}$ of length at most 3ρ in $G_{i''} := G(\downarrow Y^{i''})$ with $c'' := c(Y^{i''})$ for some $i'' \in \{0, \dots, i'-1\}$.

By continuing "unfolding" this way meta vertices closest to z, after at most h steps, we will arrive at the situation when, for some index $i^* \in \{0, 1, \dots, h\}$, a path of length at most $\rho(h+1)$ will connect vertices z and $c(Y^{i^*})$ in the graph $\widehat{G}(\downarrow Y^{i^*})$. $\square$

Consider two arbitrary vertices x and y of G, and let $S(x)$ and $S(y)$ be the nodes of $\mathcal{H}(G)$ containing x and y, respectively. Let also $NCA_{\mathcal{H}(G)}(S(x), S(y))$ be the nearest common ancestor of nodes $S(x)$ and $S(y)$ in $\mathcal{H}(G)$ and $(Y^0, Y^1, \dots, Y^h)$ be the path of $\mathcal{H}(G)$ connecting the root Y^0 of $\mathcal{H}(G)$ with $NCA_{\mathcal{H}(G)}(S(x), S(y)) = Y^h$ (in other words, $Y^0, Y^1, \dots, Y^h$ are the common ancestors of $S(x)$ and $S(y)$). Clearly, $Y^0 \cup Y^1 \cup \dots \cup Y^h$ separates vertices x and y in G.

Lemma 5. *Any path $P^G_{x,y}$ connecting vertices x and y in G contains a vertex from $Y^0 \cup Y^1 \cup \dots \cup Y^h$.*

Let $SP^G_{x,y}$ be a shortest path of G connecting vertices x and y, and let Y^i be the node of the path $(Y^0, Y^1, \dots, Y^h)$ with the smallest index such that $SP^G_{x,y} \cap Y^i \neq \emptyset$ in G. The following lemma holds.

Lemma 6. *For each $j = 0, \dots, i$, we have $d_G(x, y) = d_{G'}(x, y)$, where $G' := \widehat{G}(\downarrow Y^j)$.*

Proof. It is enough to show that the path $SP^G_{x,y}$ consists of only vertices of G'. Let assume, by a way of contradiction, that there is a vertex z of $SP^G_{x,y}$ that does not belong to G'. Let $SP^G_{x,z}$ be a subpath of $SP^G_{x,y}$ between x and z. Clearly, the node $S(z)$ of $\mathcal{H}(G)$, containing vertex z, is not a descendant of Y^i. Therefore, the nearest common ancestor of $S(x)$ and $S(z)$ in $\mathcal{H}(G)$ is a node Y^j from $\{Y^0, Y^1, \dots, Y^h\}$ with $j < i$. But then, by Lemma 5, the path $SP^G_{x,z}$ (and hence the path $SP^G_{x,y}$) must have a vertex in $Y^0 \cup Y^1 \cup \dots \cup Y^j$, contradicting with the choice of Y^i, $i > j$. $\square$

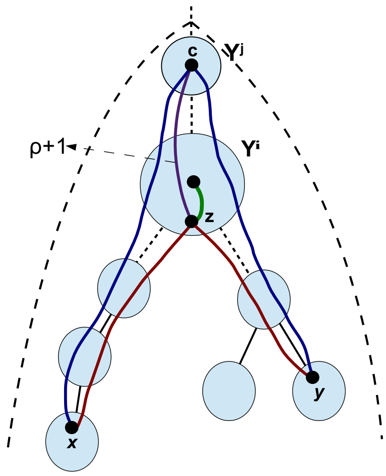


Fig. 4. Illustration to the proof of Lemma 7.

Let now $B_1^i, \ldots, B_{p_i}^i$ be the nodes at depth i of the tree $\mathcal{H}(G)$. For each node B_j^i that is not a leaf of $\mathcal{H}(G)$, consider its (central) vertex $c_j^i := c(B_j^i)$. If c_j^i is an original vertex of G (not a meta vertex created during the construction of $\mathcal{H}(G)$), then define a connected graph G_j^i obtained from $G(\downarrow B_j^i)$ by removing all its meta vertices. If removal of those meta vertices produced few connected components, choose as G_j^i that component which contains the vertex c_j^i. Denote by T_j^i a BFS-tree of graph G_j^i rooted at vertex c_j^i of B_j^i. If B_j^i is a leaf of $\mathcal{H}(G)$, then B_j^i has at most 5 vertices. In this case, remove all meta vertices from $G(\downarrow B_j^i)$ and for each connected component of the resulting graph construct an additive tree spanner with optimal surplus ≤ 3. Note that the diameter of a tree with 5 vertices is at most 4. Denote the resulting subtree (forest) by T_j^i.

The trees T_j^i ($i = 0, 1, \ldots, depth(\mathcal{H}(G))$, $j = 1, 2, \ldots, p_i$), obtained this way, are called *local subtrees* of G. Clearly, the construction of these local subtrees can be incorporated into the procedure of constructing the hierarchical tree $\mathcal{H}(G)$ of G and will not increase the overall $O(nm\log^2 n)$ run-time (see Section 3.1).

Lemma 7. *For any two vertices $x, y \in V(G)$, there exists a local subtree T such that $d_T(x, y) \le d_G(x, y) + 2\rho\log_2 n - 1$.*

Proof. We know, by Lemma 6, that a shortest path $SP_{x,y}^G$, intersecting Y^i and not intersecting any Y^l ($l < i$), lies entirely in $G' := \widehat{G}(\downarrow Y^i)$. Thus, $d_G(x, y) = d_{G'}(x, y)$. If Y^i is a leaf of $\mathcal{H}(G)$ then for a local subtree T' (it could be a forest) of G constructed for G' the following holds: $d_{T'}(x, y) \le d_{G'}(x, y) + 3 = d_G(x, y) + 3 \le d_G(x, y) + 2\rho\log_2 n - 1$ (since $n \ge 4$ and $\rho \ge 1$). Assume now that Y^i is an internal node of $\mathcal{H}(G)$. We have $i \le \log_2 n - 2$, since the depth of $\mathcal{H}(G)$ is at most $\log_2 n - 1$. Let $z \in Y^i$ be a vertex on the shortest path $SP_{x,y}^G$. By Lemma 4, there exists an index $j \in \{0, 1, \ldots, i\}$ such that the vertices z and $c(Y^j)$ can be connected in the graph $\widehat{G}(\downarrow Y^j)$ by a path of length at most $\rho(i+1)$. See Fig. 4 for an illustration. Set $G'' := \widehat{G}(\downarrow Y^j)$ and $c := c(Y^j)$. By Lemma 6, $d_G(x, y) = d_{G'}(x, y) = d_{G''}(x, y)$. Let T'' be the local tree constructed for graph $G'' = \widehat{G}(\downarrow Y^j)$, i.e., a BFS-tree of a connected component of the graph $G'' = \widehat{G}(\downarrow Y^j)$ and rooted at vertex $c = c(Y^j)$.

We have $d_{T''}(x, c) = d_{G''}(x, c)$ and $d_{T''}(y, c) = d_{G''}(y, c)$. By the triangle inequality, $d_{T''}(x, c) = d_{G''}(x, c) \le d_{G''}(x, z) + d_{G''}(z, c)$ and $d_{T''}(y, c) = d_{G''}(y, c) \le d_{G''}(y, z) + d_{G''}(z, c)$. That is, $d_{T''}(x, y) \le d_{T''}(x, c) + d_{T''}(y, c) \le d_{G''}(x, z) + d_{G''}(y, z) + 2d_{G''}(z, c) = d_{G''}(x, y) + 2d_{G''}(z, c)$. Now, using Lemma 6 and inequality $d_{G''}(z, c) \le \rho(i + 1) \le \rho(\log_2 n - 1)$, we get $d_{T''}(x, y) \le d_{G''}(x, y) + 2d_{G''}(z, c) \le d_G(x, y) + 2\rho(\log_2 n - 1)$. $\square$

This lemma implies two important results. Let G be a graph with n vertices and m edges having $\mathrm{tb}(G) \le \rho$. Also, let $\mathcal{H}(G)$ be its hierarchical tree and $\mathcal{LT}(G)$ be the family of all its local subtrees (defined above). Consider a graph H obtained by taking the union of all local subtrees of G (by putting all of them together), i.e.,

$$H := \bigcup \{T_j^i \mid T_j^i \in \mathcal{LT}(G)\} = \left(V, \cup\{E(T_j^i) \mid T_j^i \in \mathcal{LT}(G)\}\right).$$

Clearly, H is a spanning subgraph of G, constructible in $O(nm\log^2 n)$ total time, and, for any two vertices x and y of G, $d_H(x, y) \le d_G(x, y) + 2\rho\log_2 n - 1$ holds. Also, since for every level i ($i = 0, 1, \ldots, depth(\mathcal{H}(G))$) of hierarchical tree $\mathcal{H}(G)$, the corresponding local subtrees $T_1^i, \ldots, T_{p_i}^i$ are pairwise vertex-disjoint, their union has at most $n - 1$ edges. Therefore, H cannot have more than $(n - 1)\log_2 n$ edges in total. Thus, we have proven the following result.

Theorem 2. *Every graph G with n vertices and $\mathsf{tb}(G) \le \rho$ admits an additive $(2\rho \log_2 n)$-spanner with at most $n \log_2 n$ edges. Furthermore, such a sparse additive spanner of G can be constructed in polynomial time.*

Instead of taking the union of all local subtrees of G, one can fix i ($i \in \{0, 1, \dots, depth(\mathcal{H}(G))\}$) and consider separately the union of only local subtrees $T^i_1, \dots, T^i_{p_i}$, corresponding to the level i of the hierarchical tree $\mathcal{H}(G)$, and then extend in linear $O(m)$ time that forest to a spanning tree T^i of G (using, for example, a variant of the Kruskal's Spanning Tree algorithm for the unweighted graphs). We call this tree T^i the *spanning tree of G corresponding to the level i of the hierarchical tree* $\mathcal{H}(G)$. In this way we can obtain at most $\log_2 n$ spanning trees for G, one for each level i of $\mathcal{H}(G)$. Denote the collection of those spanning trees by $\mathcal{T}(G)$. Thus, we obtain the following theorem.

Theorem 3. *Every graph G with n vertices and $\mathsf{tb}(G) \le \rho$ admits a system $\mathcal{T}(G)$ of at most $\log_2 n$ collective additive tree $(2\rho \log_2 n)$-spanners. Furthermore, such a system of collective additive tree spanners of G can be constructed in polynomial time.*

3.3. Additive spanners for graphs admitting (multiplicative) tree t-spanners

Now we give two implications of the above results for the class of tree t-spanner admissible graphs. In [28], the following important ("bridging") lemma was proven.

Lemma 8. *(See [28].) If a graph G admits a tree t-spanner then its tree-breadth is at most $\lceil t/2 \rceil$.*

Note that the tree-breadth bounded by $\lceil t/2 \rceil$ provides only a necessary condition for a graph to have a multiplicative tree t-spanner. There are (chordal) graphs which have tree-breadth 1 but any multiplicative tree t-spanner of them has $t = \Omega(\log n)$ [28]. Furthermore, a cycle on $3n$ vertices has tree-breadth n but admits a system of 2 collective additive tree 0-spanners.

Combining Lemma 8 with Theorem 2 and Theorem 3, we deduce the following results.

Theorem 4. *Let G be a graph with n vertices and m edges having a (multiplicative) tree t-spanner. Then, G admits an additive $(2\lceil t/2 \rceil \log_2 n)$-spanner with at most $n \log_2 n$ edges constructible in $O(nm \log^2 n)$ time.*

Theorem 5. *Let G be a graph with n vertices and m edges having a (multiplicative) tree t-spanner. Then, G admits a system $\mathcal{T}(G)$ of at most $\log_2 n$ collective additive tree $(2\lceil t/2 \rceil \log_2 n)$-spanners constructible in $O(nm \log^2 n)$ time.*

4. Collective additive tree spanners of graphs with bounded k-tree-breadth, $k \ge 2$

In this section, we extend the approach of Section 3 and show that any n-vertex graph G with $\mathsf{tb}_k(G) \le \rho$ has a system of at most $k(1 + \log_2 n)$ collective additive tree $(2\rho(1 + \log_2 n))$-spanners constructible in polynomial time for every fixed k. We will assume that $n > k$, since any graph with n vertices has a system of $n - 1$ collective additive tree 0-spanners (consider $n - 1$ BFS-trees rooted at different vertices).

4.1. Balanced separators for graphs with bounded k-tree-breadth

We will need the following balanced clique-separator result for chordal graphs. Recall that a graph is chordal if each of its induced cycles has length three.

Theorem 6. *(See [44].) Every chordal graph G with n vertices and m edges contains a maximal clique C such that if the vertices in C are deleted from G, every connected component in the graph induced by any remaining vertices is of size at most $n/2$. Such a balanced clique-separator C of a connected chordal graph can be found in $O(m)$ time.*

We say that a graph $G = (V, E)$ with $|V| \ge k$ has a *balanced* $\mathbf{D^k_r}$*-separator* if there exists a collection of k disks $D_r(v_1, G), D_r(v_2, G), \dots, D_r(v_k, G)$ in G, centered at (different) vertices $v_1, v_2, \dots, v_k$ and each of radius r, such that the union of those disks $\mathbf{D^k_r} := \bigcup_{i=1}^k D_r(v_i, G)$ forms a balanced separator of G, i.e., each connected component of $G[V \setminus \mathbf{D^k_r}]$ has at most $|V|/2$ vertices. The following result generalizes Lemma 1.

Lemma 9. *Every graph G with at least k vertices and $\mathsf{tb}_k(G) \le \rho$ has a* balanced $\mathbf{D^k_\rho}$-separator.

Proof. The proof of this lemma follows from *acyclic hypergraph* theory. First we review some necessary definitions and an important result characterizing acyclic hypergraphs. Recall that a *hypergraph* H is a pair $H = (V, \mathcal{E})$ where V is a set of vertices and $\mathcal{E}$ is a set of non-empty subsets of V called *hyperedges*. For these and other hypergraph notions see [10].

Let $H = (V, \mathcal{E})$ be a *hypergraph* with the vertex set V and the *hyperedge* set $\mathcal{E}$. For every vertex $v \in V$, let $\mathcal{E}(v) = \{e \in \mathcal{E} | v \in e\}$. The *2-section graph* $2SEC(H)$ of a hypergraph H has V as its vertex set and two distinct vertices are adjacent in $2SEC(H)$ if and only if they are contained in a common hyperedge of H. A hypergraph H is called *conformal* if every clique of $2SEC(H)$ is contained in a hyperedge $e \in \mathcal{E}$, and a hypergraph H is called *acyclic* if there is a tree T with node set $\mathcal{E}$ such that for all vertices $v \in V$, $\mathcal{E}(v)$ induces a subtree T_v of T. It is a well-known fact (see, e.g., [3,9,10]) that a hypergraph H is acyclic if and only if H is conformal and $2SEC(H)$ of H is a chordal graph.

Let now $G = (V, E)$ be a graph with $\mathrm{tb}_k(G) = \rho$ and $T(G) = (\{X_i | i \in I\}, T = (I, F))$ be its tree-decomposition of k-breadth ρ. Evidently, the third condition of tree-decompositions can be restated as follows: the hypergraph $H = (V(G), \{X_i | i \in I\})$ is an acyclic hypergraph. Since each edge of G is contained in at least one bag of $T(G)$, the 2-section graph $G^* := 2SEC(H)$ of H is a chordal *supergraph* of the graph G (each edge of G is an edge of G^*, but G^* may have some extra edges between non-adjacent vertices of G contained in a common bag of $T(G)$). By Theorem 6, the chordal graph G^* contains a balanced clique-separator $C \subseteq V(G)$. By conformality of H, C must be contained in a bag of $T(G)$. From the definition of k-breadth, there must exist k vertices $v_1, v_2, \ldots, v_k$ such that $C \subseteq \mathbf{D}^{\mathbf{k}}_{\rho}$, where $\mathbf{D}^{\mathbf{k}}_{\rho} = D_\rho(v_1, G) \cup \cdots \cup D_\rho(v_k, G)$. As the removal of the vertices of C from G^* leaves no connected component in $G^*[V \setminus C]$ with more than $|V|/2$ vertices and since G^* is a supergraph of G, clearly, the removal of the vertices of $\mathbf{D}^{\mathbf{k}}_{\rho}$ from G leaves no connected component in $G[V \setminus \mathbf{D}^{\mathbf{k}}_{\rho}]$ with more than $|V|/2$ vertices. □

Again, like in Section 3, we do not assume that a tree-decomposition $T(G)$ of k-breadth ρ is given for G as part of the input. Our method does not need to know $T(G)$. For a given graph G, integers $k \geq 1$ and $\rho \geq 0$, even checking whether G has a tree-decomposition of k-breadth ρ is a hard problem (as $\mathrm{tb}_k(G) = 0$ if and only if $\mathrm{tw}(G) \leq k - 1$) (see Section 1.2).

Let G be an arbitrary connected n-vertex m-edge graph. In [28], an algorithm was described which, given G and its arbitrary fixed vertex v, finds in $O(m)$ time a balanced disk separator $D_r(v, G)$ of G centered at v and with minimum r. We can use this algorithm as a subroutine to find for G in $O(n^k m)$ time a balanced $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}}$-separator with minimum r. Given arbitrary k vertices $v_1, v_2, \ldots, v_k$ of G, we can add a new dummy vertex x to G and make it adjacent to only $v_1, v_2, \ldots, v_k$ in G. Denote the resulting graph by $G + x$. Then, a balanced disk separator $D_{r+1}(x, G + x)$ of $G + x$ with minimum $r + 1$ gives a balanced separator of G of the form $D_r(v_1, G) \cup \cdots \cup D_r(v_k, G)$ (for particular disk centers $v_1, v_2, \ldots, v_k$) with minimum r. Iterating over all k vertices of G, we can find a balanced $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}}$-separator of G with the smallest (absolute minimum) radius r. Thus, we have the following result.

Proposition 3. *Let k be a positive integer (assumed to be small). For an arbitrary graph G with $n \geq k$ vertices and m edges, a balanced $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}}$-separator with the smallest radius r can be found in $O(n^k m)$ time.*

4.2. Decomposition of a graph with bounded k-tree-breadth

Let $G = (V, E)$ be an arbitrary connected graph with n vertices and m edges and with a balanced $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}}$-separator, where $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}} = \bigcup_{j=1}^{k} D_r(v_j, G)$. Note that some disks in $\{D_r(v_1, G), \ldots, D_r(v_k, G)\}$ may overlap. In what follows, we will partition $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}} = \bigcup_{j=1}^{k} D_r(v_j, G)$ into k sets $D_1, \ldots, D_k$ such that no two of them intersect and each D_j, $j = 1, \ldots, k$, contains at least one vertex v_j and induces a connected subgraph of $G[D_r(v_j, G)]$. Create a graph $G + s$ by adding a new dummy vertex s to G and making it adjacent to only $v_1, v_2, \ldots, v_k$ in G. Let T be a BFS-tree of $G + s$ started at vertex s and T' be a subtree of T formed by vertices $\{v \in V(G + s) | d_T(s, v) \leq r + 1\}$ and rooted at s. Let also $T(v_1), \ldots, T(v_k)$ be the subtrees of $T' \setminus \{s\}$ rooted at $v_1, \ldots, v_k$, respectively. Clearly, each $T(v_j)$, $j = 1, \ldots, k$, is a subtree (not necessarily spanning) of $G[D_r(v_j, G)]$ and $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}} = \bigcup_{j=1}^{k} V(T(v_j))$. Set now $D_j := V(T(v_j))$, $j = 1, \ldots, k$.

Let $G_1, G_2, \ldots, G_q$ be the connected components of $G[V \setminus \mathbf{D}^{\mathbf{k}}_{\mathbf{r}}]$. Denote by $S_i^j = \{v \in V(G_i) | d_G(v, D_j) = 1\}$, $i = 1, \ldots, q$, $j = 1, \ldots, k$, the neighborhood of D_j in G_i. Also, let G_i^+ be the graph obtained from component G_i by adding one *meta vertex* c_i^j for each disk $D_r(v_j, G)$ (a *representative of* $D_r(v_j, G)$), $j = 1, \ldots k$, and making it adjacent to all vertices of S_i^j, i.e., for a vertex $x \in V(G_i)$, $c_i^j x \in E(G_i^+)$ if and only if there is a vertex $x_D \in D_j \subseteq D_r(v_j, G)$ with $x x_D \in E(G)$. If S_i^j is empty for some j, then vertex c_i^j is not added to G_i^+. Also, add an edge between any two representatives c_i^j and c_i^l if vertices v_j and v_l are connected by a path in $G[V \setminus V(G_i)]$. See Fig. 5 for an illustration.

Given an n-vertex m-edge graph G and its balanced $\mathbf{D}^{\mathbf{k}}_{\mathbf{r}}$-separator, the graphs $G_1^+, \ldots, G_q^+$ can be constructed in total time $O(kqm)$. Furthermore, the total number of edges in graphs $G_1^+, \ldots, G_q^+$ does not exceed $m + qk^2$, and the total number of vertices in those graphs does not exceed the number of vertices in $G[V \setminus \mathbf{D}^{\mathbf{k}}_{\mathbf{r}}]$ plus qk.

Note that G_i^+ is a minor of G and can be obtained from G by a sequence of edge contractions in the following way. First contract all edges (in any order) that are incident to $V(G_{i'})$, for all $i' = 1, \ldots, q$, $i' \neq i$. Then, for each $j = 1, \ldots, k$, contract (all edges of) connected subgraph $G[D_j]$ of G to get meta vertex c_i^j representing the disk $D_r(v_j, G)$ in G_i^+.

Let again $G_{/e}$ be the graph obtained from G by contracting edge e. We have the following analog of Lemma 3.

Lemma 10. *For any graph G and its edge e, $\mathrm{tb}_k(G) \leq \rho$ implies $\mathrm{tb}_k(G_{/e}) \leq \rho$. Consequently, for any graph G with $\mathrm{tb}_k(G) \leq \rho$, $\mathrm{tb}_k(G_i^+) \leq \rho$ holds for $i = 1, \ldots, q$.*

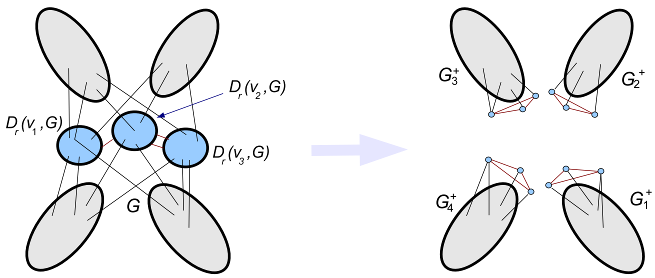


Fig. 5. A graph G with a balanced $\mathbf{D_r^3}$-separator and the corresponding graphs $G_1^+,\dots,G_4^+$ obtained from G. Each G_i^+ has three meta vertices representing the three disks.

Proof. Our proof is similar to the proof from [28] of Lemma 3. We provide it here for the sake of completeness. Let $T(G)=(\{X_i|i\in I\},T=(I,F))$ be a tree-decomposition of G with k-breadth ρ. Let $e=xy$ be an arbitrary edge of G. We can obtain a tree-decomposition $T(G_{/e})$ of the graph $G_{/e}$ by replacing in each bag X_i, $i\in I$, vertices x and y with a new vertex x' representing them (if some bag A contained both x and y, only one copy of x' is kept). Evidently, the first and the second conditions of tree-decompositions are fulfilled for $T(G_{/e})$. Furthermore, the topology (the tree $T=(I,F)$) of the tree-decomposition did not change. Still, for any vertex $v\neq x'$ of $G_{/e}$, the bags of $T(G_{/e})$ containing v form a subtree in $T(G_{/e})$. Since vertices x and y were adjacent in G, there was a bag A of $T(G)$ containing both those vertices. Hence, a subtree of $T(G_{/e})$ formed by bags of $T(G_{/e})$ containing vertex x' is nothing else but the union of two subtrees (one for x and one for y) of $T(G)$ sharing at least one common bag A. Also, contracting an edge can only reduce the distances in a graph. Hence, still, for each bag B of $T(G_{/e})$, there must exist corresponding vertices $v_1,\dots,v_k$ in $G_{/e}$ with $B\subseteq D_\rho(v_1,G_{/e})\cup\dots\cup D_\rho(v_k,G_{/e})$. Thus, $\mathrm{tb}_k(G_{/e})\le\rho$. Since G_i^+ can be obtained from G by a sequence of edge contractions, we also have $\mathrm{tb}_k(G_i^+)\le\rho$. $\square$

4.3. Construction of a hierarchical tree

Here we show how a hierarchical tree for a graph with bounded k-tree-breadth is built.

Let $G=(V,E)$ be a connected n-vertex, m-edges graph with $\mathrm{tb}_k(G)\le\rho$ and $n\ge k$. Lemma 9 guaranties that G has a balanced $\mathbf{D_r^k}$-separator with $r\le\rho$. Proposition 3 says that such a balanced $\mathbf{D_r^k}$-separator of G can be found in $O(n^k m)$ time by an algorithm that works directly on the graph G and does not require construction of a tree-decomposition of G with k-breadth $\le\rho$. Using these and Lemma 10, we can build a rooted *hierarchical-tree* $\mathcal{H}(G)$ for G, which is constructed as follows. If G is a connected graph with at most $2k+1$ vertices, then $\mathcal{H}(G)$ is a one node tree with root node $(V(G),G)$. It is known [46] that any connected graph with $p\ge 2$ vertices has a dominating set of size $\lfloor p/2\rfloor$, i.e., all vertices of it can be covered by $\lfloor p/2\rfloor$ disks of radius one. Hence, in our case, G with at most $2k+1$ vertices can be covered by k disks of radius one each, i.e., there are k vertices $v_1,\dots,v_k$ such that $V(G)=D_r(v_1,G)\cup\dots\cup D_r(v_k,G)$ for $r=1\le\rho$. If G is a connected graph with more than $2k+1$ vertices, find a balanced $\mathbf{D_r^k}$-separator of minimum radius r in $O(n^k m)$ time and construct the corresponding graphs $G_1^+,\dots,G_q^+$. For each graph G_i^+, $i\in\{1,\dots,q\}$, (by Lemma 10, $\mathrm{tb}_k(G_i^+)\le\rho$) construct a hierarchical tree $\mathcal{H}(G_i^+)$ recursively and build $\mathcal{H}(G)$ by taking the pair $(\mathbf{D_r^k},G)$ to be the root and connecting the root of each tree $\mathcal{H}(G_i^+)$ as a child of $(\mathbf{D_r^k},G)$.

The depth of this tree $\mathcal{H}(G)$ is the smallest integer p such that

$$\frac{n}{2^p}+k\left(\frac{1}{2^{p-1}}+\dots+\frac{1}{2}+1\right)\le 2k+1,$$

that is, the depth is at most $\log_2 n$. It is also not hard to see that, given a graph G with n vertices and m edges, a hierarchical tree $\mathcal{H}(G)$ can be constructed in $O((kn)^{k+2}\log^{k+1}n)$ total time. There are at most $O(\log n)$ levels in $\mathcal{H}(G)$, and one needs to do at most $O((n+kn\log n)^k(m+k^2 n\log n))\le O((kn)^{k+2}\log^k n)$ operations per level since the total number of edges in the graphs of each level is at most $O(m+k^2n\log n)$ and the total number of vertices in those graphs cannot exceed $O(n+kn\log n)$.

For nodes of $\mathcal{H}(G)$, we use the same notations as in Section 3. For a node Y of $\mathcal{H}(G)$, since it is associated with a pair $(\mathbf{D_{r'}^k},G')$, where $r'\le\rho$, G' is a minor of G and $\mathbf{D_{r'}^k}=D_{r'}(v_1',G')\cup\dots\cup D_{r'}(v_1',G')$, it is convenient to denote G' by $G(\downarrow Y)$, $\{v_1',\dots,v_k'\}$ by $c(Y)=\{c_1(Y),\dots,c_k(Y)\}$, r' by $r(Y)$, and $\mathbf{D_{r'}^k}$ by Y itself. Thus, $(\mathbf{D_{r'}^k},G')=(\bigcup_{l=1}^k D_{r(Y)}(c_l(Y),G(\downarrow Y)),G(\downarrow Y))=(Y,G(\downarrow Y))$ in these notations, and we identify node Y of $\mathcal{H}(G)$ with the set $\bigcup_{l=1}^k D_{r(Y)}(c_l(Y),G(\downarrow Y))$ and associate with this node also the graph $G(\downarrow Y)$. If now $(Y^0,Y^1,\dots,Y^h)$ is the path of $\mathcal{H}(G)$ connecting the root Y^0 of $\mathcal{H}(G)$ with a node Y^h, then the vertex set of the graph $G(\downarrow Y^h)$ consists of some (original) vertices of G plus at most kh meta vertices representing the disks $D_{r(Y)}(c_1(Y^i),G(\downarrow Y^i)),\dots,D_{r(Y)}(c_k(Y^i),G(\downarrow Y^i))$ of Y^i, $i=0,1,\dots,h-1$. Note also that each (original) vertex of G belongs to exactly one node of $\mathcal{H}(G)$.

4.4. Construction of collective additive tree spanners

Let $G=(V,E)$ be a connected n-vertex, m-edge graph and assume that $\mathrm{tb}_k(G)\le\rho$ and $n\ge k$. Let $\mathcal{H}(G)$ be a hierarchical tree of G. Consider an arbitrary node Y^h of $\mathcal{H}(G)$, and let $(Y^0,Y^1,\dots,Y^h)$ be the path of $\mathcal{H}(G)$ connecting the root Y^0 of $\mathcal{H}(G)$ with Y^h. Let $\widehat{G}(\downarrow Y^j)$ be the graph obtained from $G(\downarrow Y^j)$ by removing all its meta vertices (note that $\widehat{G}(\downarrow Y^j)$ may be disconnected and that all meta vertices of $G(\downarrow Y^j)$ come from previous levels of $\mathcal{H}(G)$). We have the following analog of Lemma 4.

Lemma 11. *For any vertex z from $Y^h\cap V(G)$ there exists an index $i\in\{0,1,\dots,h\}$ such that the vertices z and $c_l(Y^i)$, for some $l\in\{1,\dots,k\}$ can be connected in the graph $\widehat{G}(\downarrow Y^i)$ by a path of length at most $\rho(h+1)$. In particular, $d_G(z,c_l(Y^i))\le\rho(h+1)$ holds.*

Proof. The proof is similar to the proof of Lemma 4 of Section 3. Set $G_h:=G(\downarrow Y^h)$ and $c:=c_l(Y^h)$, where $z\in D_l\subseteq D_{r(Y^h)}(c_l(Y^h),G_h)$ (for the definition of set D_l see the first paragraph of Section 4.2). Let $SP^{G_h}_{c,z}$ be a shortest path of G_h connecting vertices c and z. We know that this path has at most $r(Y^h)\le\rho$ edges. If $SP^{G_h}_{c,z}$ does not contain any meta vertices, then this path is a path of $\widehat{G}(\downarrow Y^h)$ and of G and therefore $d_G(c,z)\le\rho$ holds.

Assume now that $SP^{G_h}_{c,z}$ does contain meta vertices and let μ' be the closest to z meta vertex in $SP^{G_h}_{c,z}$ (consult with Fig. 3). Let $SP^{G_h}_{c,z}=(c,\dots,a',\mu',b',\dots,z)$. By construction of $\mathcal{H}(G)$, meta vertex μ' was created at some earlier recursive step to represent one disk of $Y^{i'}$ of graph $G_{i'}:=G(\downarrow Y^{i'})$ for some $i'\in\{0,\dots,h-1\}$. Hence, there is a path $P^{G_{i'}}_{c',z}=(c',\dots,b',\dots,z)$ of length at most 2ρ in $G_{i'}$ with $c':=c_{l'}(Y^{i'})$ for some $l'\in\{1,\dots,k\}$. Again, if $P^{G_{i'}}_{c',z}$ does not contain any meta vertices, then this path is a path of $\widehat{G}(\downarrow Y^{i'})$ and of G and therefore $d_G(c',z)\le 2\rho$ holds. If $P^{G_{i'}}_{c',z}$ does contain meta vertices then again, "unfolding" a meta vertex μ'' of $P^{G_{i'}}_{c',z}$ closest to z, we obtain a path $P^{G_{i''}}_{c'',z}$ of length at most 3ρ in $G_{i''}:=G(\downarrow Y^{i''})$ with $c'':=c_{l''}(Y^{i''})$ for some $i''\in\{0,\dots,i'-1\}$ and $l''\in\{1,\dots,k\}$.

We continue "unfolding" this way meta vertices closest to z. Eventually, after at most h steps, we will arrive at the situation when, for some index $i^*\in\{0,1,\dots,h\}$, a path of length at most $\rho(h+1)$ will connect vertices z and $c_{l^*}(Y^{i^*})$, for some $l^*\in\{1,\dots,k\}$, in the graph $\widehat{G}(\downarrow Y^{i^*})$. □

Let $B^i_1,\dots,B^i_{p_i}$ be the nodes at depth i of the tree $\mathcal{H}(G)$. Assume $B^i_j=\bigcup_{l=1}^k D_r(c^i_j(l),G(\downarrow B^i_j))$, where $r:=r(B^i_j)$. Denote k central vertices of B^i_j by $C^i_j=\{c^i_j(1),c^i_j(2),\dots,c^i_j(k)\}$. For each node B^i_j, consider its (central) vertex $c^i_j(l)$ ($l\in\{1,\dots,k\}$). If $c^i_j(l)$ is an original vertex of G (not a meta vertex created during the construction of $\mathcal{H}(G)$), then define a connected graph $G^i_j(l)$ obtained from $G(\downarrow B^i_j)$ by removing all its meta vertices. If removal of those meta vertices produced few connected components, choose as $G^i_j(l)$ that component which contains the vertex $c^i_j(l)$. Denote by $T^i_j(l)$ a BFS-tree of graph $G^i_j(l)$ rooted at vertex $c^i_j(l)$ of B^i_j.

The trees $T^i_j(l)$ ($i=0,1,\dots,depth(\mathcal{H}(G))$, $j=1,2,\dots,p_i$, $l=1,2,\dots,k$), obtained this way, are called *local subtrees* of G. Clearly, the construction of these local subtrees can be incorporated into the procedure of constructing a hierarchical tree $\mathcal{H}(G)$ of G and will not increase the overall $O((kn)^{k+2}\log^{k+1}n)$ run-time (see Section 4.3).

Since Lemma 5 and Lemma 6 hold for G, similarly to the proof of Lemma 7, one can prove its analog for graphs with bounded k-tree-breadth.

Lemma 12. *For any two vertices $x,y\in V(G)$, there exists a local subtree T such that $d_T(x,y)\le d_G(x,y)+2\rho(1+\log_2 n)$.*

This lemma implies the following two results. Let G be a graph with n vertices and m edges having $\mathrm{tb}_k(G)\le\rho$. Let also $\mathcal{H}(G)$ be its hierarchical tree and $\mathcal{LT}(G)$ be the family of all its local subtrees (defined above). Consider a graph H obtained by taking the union of all local subtrees of G (by putting all of them together). Clearly, H is a spanning subgraph of G, constructible in polynomial time for every fixed k. We have $d_H(x,y)\le d_G(x,y)+2\rho(1+\log_2 n)$ for any two vertices x and y of G. Also, since for every level i ($i=0,1,\dots,depth(\mathcal{H}(G))$) of hierarchical tree $\mathcal{H}(G)$, the corresponding local subtrees $T^i_1(l),\dots,T^i_{p_i}(l)$ for each fixed index $l\in\{1,\dots,k\}$ are pairwise vertex-disjoint, their union has at most $n-1$ edges. Therefore, H cannot have more than $k(n-1)(1+\log_2 n)$ edges in total. Thus, we have the following result.

Theorem 7. *Every graph G with n vertices and $\mathrm{tb}_k(G)\le\rho$ admits an additive $(2\rho(1+\log_2 n))$-spanner with at most $O(kn\log n)$ edges constructible in polynomial time for every fixed k.*

For a node B^i_j of $\mathcal{H}(G)$, let $\mathcal{T}^i_j=\{T^i_j(1),\dots,T^i_j(k)\}$ be the set of its local subtrees. Instead of taking the union of all local subtrees of G, one can fix i ($i\in\{0,1,\dots,depth(\mathcal{H}(G))\}$) and fix $l\in\{1,\dots,k\}$ and consider separately the union of only local subtrees $T^i_1(l),\dots,T^i_{p_i}(l)$, corresponding to the lth subtrees of level i of the hierarchical tree $\mathcal{H}(G)$, and then extend in linear $O(m)$ time that forest to a spanning tree $T^i(l)$ of G (using, for example, a variant of the Kruskal's Spanning Tree algorithm for the unweighted graphs). We call this tree $T^i(l)$ the lth *spanning tree of G corresponding to the level i of the hierarchical tree* $\mathcal{H}(G)$. In this way we can obtain at most $k(1+\log_2 n)$ spanning trees for G, k trees for each level i of $\mathcal{H}(G)$. Denote the collection of those spanning trees by $\mathcal{T}(G)$. Thus, we deduce the following theorem.

Theorem 8. *Every graph G with n vertices and $\mathrm{tb}_k(G) \le \rho$ admits a system $\mathcal{T}(G)$ of at most $k(1+\log_2 n)$ collective additive tree $(2\rho(1+\log_2 n))$-spanners constructible in polynomial time for every fixed k.*

5. Additive spanners for graphs admitting (multiplicative) *t*-spanners of bounded tree-width

In this section, we show that if a graph G admits a (multiplicative) t-spanner H with $\mathrm{tw}(H) = k-1$ then its k-tree-breadth is at most $\lceil t/2 \rceil$. As a consequence, we obtain that, for every fixed k, there is a polynomial time algorithm that, given an n-vertex graph G admitting a (multiplicative) t-spanner with tree-width at most $k-1$, constructs a system of at most $k(1+\log_2 n)$ collective additive tree $O(t \log n)$-spanners of G.

5.1. k-Tree-breadth of a graph admitting a t-spanner of bounded tree-width

Let H be a graph with tree-width $k-1$, and let $T(H) = (\{X_i | i \in I\}, T = (I, F))$ be its tree-decomposition of width $k-1$. For an integer $r \ge 0$, denote by $X_i^{(r)}$, $i \in I$, the set $D_r(X_i, H) := \bigcup_{x \in X_i} D_r(x, H)$. Clearly, $X_i^{(0)} = X_i$ for every $i \in I$. The following important lemma holds.

Lemma 13. *For every integer $r \ge 0$, $T^{(r)}(H) := (\{X_i^{(r)} | i \in I\}, T = (I, F))$ is a tree-decomposition of H with k-breadth $\le r$.*

Proof. It is enough to show that the third condition of tree-decompositions (see Section 1.2) is fulfilled for $T^{(r)}(H)$. That is, for all $i, j, k \in I$, if j is on the path from i to k in T, then $X_i^{(r)} \cap X_k^{(r)} \subseteq X_j^{(r)}$. We know that $X_i \cap X_k \subseteq X_j$ holds and need to show that for every vertex v of H, $d_H(v, X_i) \le r$ and $d_H(v, X_k) \le r$ imply $d_H(v, X_j) \le r$. Assume, by way of contradiction, that for some integer $r > 0$ and for some vertex v of H, $d_H(v, X_j) > r$ while $d_H(v, X_i) \le r$ and $d_H(v, X_k) \le r$.

Consider the original tree-decomposition $T(H)$. It is known [21] that if ab $(a, b \in I)$ is an edge of the tree $T = (I, F)$ of tree-decomposition $T(H)$, and T_a, T_b are the subtrees of T obtained after removing edge ab from T, then $S = X_a \cap X_b$ separates in H vertices belonging to bags of T_a but not to S from vertices belonging to bags of T_b but not to S. We will use this nice separation property.

Let $T \setminus \{j\}$ be the forest obtained from T by removing node j, and let $T(i)$ and $T(k)$ be the trees from this forest containing nodes i and k, respectively. Clearly, $T(i)$ and $T(k)$ are disjoint. The above separation property and inequalities $d_H(v, X_i) \le r < d_H(v, X_j)$ ensure that the vertex v belongs to a node (a bag) of $T(i)$ (X_j cannot separate in H vertex v from a vertex x_i of X_i with $d_H(v, X_i) = d_H(v, x_i)$ since otherwise $d_H(v, X_i) > d_H(v, X_j)$ will hold). Similarly, inequalities $d_H(v, X_k) \le r < d_H(v, X_j)$ and the above separation property guarantee that the vertex v belongs to a node of $T(k)$. But then, the third condition of tree-decompositions says that v must also belong to the bag X_j of $T(H)$. The latter, however, is in a contradiction to the assumption that $d_H(v, X_j) > r \ge 0$. □

Now we can prove the main lemma of this section.

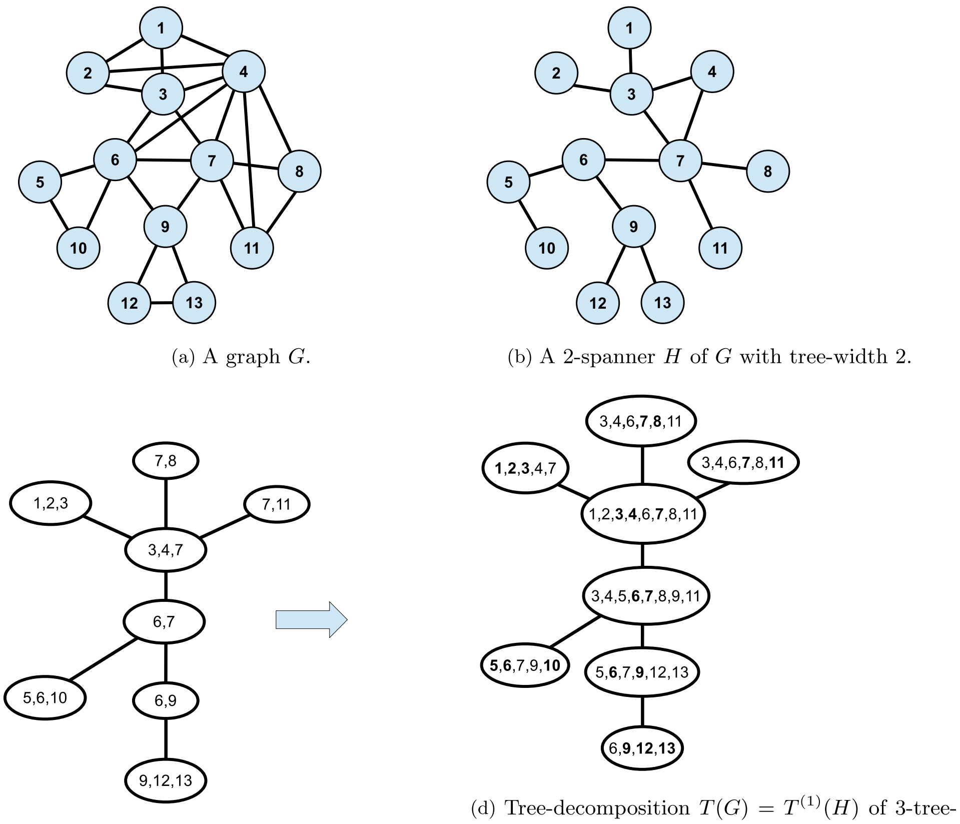
Lemma 14. *If a graph G admits a t-spanner with tree-width $k-1$, then $\mathrm{tb}_k(G) \le \lceil t/2 \rceil$.*

Proof. Let H be a t-spanner of G with $\mathrm{tw}(G) = k-1$ and $T(H) = (\{X_i | i \in I\}, T = (I, F))$ be a tree-decomposition of H of width $k-1$. We claim that $T(G) := T^{(\lceil t/2 \rceil)}(H) := (\{X_i^{(\lceil t/2 \rceil)} | i \in I\}, T = (I, F))$ is a tree-decomposition of G with k-breadth $\le \lceil t/2 \rceil$. See Fig. 6 for an illustration.

By Lemma 13, $T^{(\lceil t/2 \rceil)}(H)$ is a tree-decomposition of H with k-breadth $\le \lceil t/2 \rceil$. Hence, the first and the third conditions of tree-decompositions hold for $T(G)$. For every pair u, v of vertices of G, $d_G(u, v) \le d_H(u, v)$. Therefore, every disk $D_{\lceil t/2 \rceil}(x, H)$ of H is contained in a disk $D_{\lceil t/2 \rceil}(x, G)$ of G. This implies that every bag of $T(G)$ is covered by at most k disks of G of radius at most $\lceil t/2 \rceil$ each, i.e.,

$$X_i^{(\lceil t/2 \rceil)} = D_{\lceil t/2 \rceil}(X_i, H) = \bigcup_{x \in X_i} D_{\lceil t/2 \rceil}(x, H) \subseteq \bigcup_{x \in X_i} D_{\lceil t/2 \rceil}(x, G).$$

We need only to show additionally that each edge uv of G belongs to some bag of $T(G)$. Since H is a t-spanner of G, $d_H(u, v) \le t$ holds. Let x be a middle vertex of a shortest path connecting u and v in H. Then, both u and v belong to the disk $D_{\lceil t/2 \rceil}(x, H)$. Let X_i be a bag of $T(H)$ containing vertex x. Then, both u and v are contained in $X_i^{(\lceil t/2 \rceil)}$, a bag of $T(G)$. □



(a) A graph G.

(b) A 2-spanner H of G with tree-width 2.

(c) Tree-decomposition $T(H)$ of width 2.

(d) Tree-decomposition $T(G) = T^{(1)}(H)$ of 3-tree-breadth equal 1.

Fig. 6. Illustration to the proof of Lemma 14. A tree-decomposition for G is obtained from a tree-decomposition of H.

5.2. Consequences

Now we give two implications of the above results for the class of graphs admitting (multiplicative) t-spanners with tree-width $k-1$. They are direct consequences of Lemma 14, Theorem 7 and Theorem 8.

Theorem 9. *Let G be a graph with n vertices and m edges having a (multiplicative) t-spanner with tree-width $k-1$. Then, G admits an additive $(2\lceil t/2\rceil(1+\log_2 n))$-spanner with at most $O(kn\log n)$ edges constructible in polynomial time for every fixed k.*

Theorem 10. *Let G be a graph with n vertices and m edges having a (multiplicative) t-spanner with tree-width $k-1$. Then, G admits a system $\mathcal{T}(G)$ of at most $k(1+\log_2 n)$ collective additive tree $(2\lceil t/2\rceil(1+\log_2 n))$-spanners constructible in polynomial time for every fixed k.*

6. Concluding remarks and open problems

Using Robertson–Seymour's tree-decomposition of graphs, we described a necessary condition for a graph to have a multiplicative t-spanner of tree-width k (in particular, to have a multiplicative tree t-spanner, when $k=1$). As we have mentioned earlier, this necessary condition is far from being sufficient. The following interesting problem remains open.

– Does there exist a clean "if and only if" condition under which a graph admits a multiplicative (or, additive) t-spanner of tree-width k (in particular, admits a multiplicative (or, additive) tree t-spanner ($k=1$ case))?

That necessary condition was very useful in demonstrating that, for every fixed k, there is a polynomial time algorithm that, given an n-vertex graph G admitting a multiplicative t-spanner with tree-width k, constructs a system of at most $(k+1)(1+\log_2 n)$ collective additive tree $O(t\log n)$-spanners of G. In particular, when $k=1$, we showed that there is a polynomial time algorithm that, given an n-vertex graph G admitting a multiplicative tree t-spanner, constructs a system of at most $\log_2 n$ collective additive tree $O(t\log n)$-spanners of G. Can these results be improved?

– Does a polynomial time algorithm exist that, given an n-vertex graph G admitting a multiplicative tree t-spanner, constructs a system of $O(1)$ collective additive tree $O(t)$-spanners of G?
– Does a polynomial time algorithm exist that, given an n-vertex graph G admitting a multiplicative t-spanner with tree-width k, constructs a system of $O(k)$ collective additive tree $O(t)$-spanners of G?

As we have mentioned earlier, an interesting particular question whether a multiplicative tree spanner can be turned in polynomial time into a (one) additive tree spanner with a slight increase in the stretch is (negatively) settled already in [40]. Yet, it is interesting to know whether an exponential time procedure that performs such a transformation exists.

Two more interesting challenging questions we leave for future investigation.

– Is there any polynomial time algorithm which, given a graph admitting a system of at most μ collective tree t-spanners, constructs a system of at most $\alpha(\mu, n)$ collective tree $\beta(t, n)$-spanners, where $\alpha(\mu, n)$ is $O(\mu)$ (or $O(\mu \log n)$) and $\beta(t, n)$ is $O(t)$ (or $O(t \log n)$)?

In this approximation question, we assume that one knows that a graph G admits a system of at most μ collective tree t-spanners, but (s)he does not know how to find it in polynomial time and wonders if something weaker can be constructed efficiently. The following question is about approximating the k-TREE-WIDTH t-SPANNER problem.

– Is there a polynomial time algorithm that, for every unweighted graph G admitting a t-spanner of tree-width k, constructs an $(O(k \log n)t)$-spanner with tree-width at most k?

Acknowledgements

We would like to thank anonymous reviewers for many useful suggestions and comments.

References

[1] N. Alon, R.M. Karp, D. Peleg, D.B. West, A graph-theoretic game and its application to the k-server problem, SIAM J. Comput. 24 (1995) 78–100.
[2] I. Althöfer, G. Das, D. Dobkin, D. Joseph, J. Soares, On sparse spanners of weighted graphs, Discrete Comput. Geom. 9 (1) (1993) 81–100.
[3] G. Ausiello, A. D'Arti, M. Moscarini, Chordality properties on graphs and minimal conceptual connections in semantic data models, J. Comput. System Sci. 33 (1986) 179–202.
[4] Y. Bartal, Probabilistic approximations of metric spaces and its algorithmic applications, in: Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 1996, pp. 184–193.
[5] Y. Bartal, On approximating arbitrary metrics by tree metrics, in: Proceedings of the 30th Annual ACM Symposium on Theory of Computing, 1998, pp. 161–168.
[6] Y. Bartal, A. Blum, C. Burch, A. Tomkins, A polylog()-competitive algorithm for metrical task systems, in: STOC, 1997, pp. 711–719.
[7] S. Baswana, S. Sen, A simple linear time algorithm for computing a $(2k-1)$-spanner of $O(n^{1+1/k})$ size in weighted graphs, in: 30th International Colloquium on Automata, Languages and Programming, ICALP, in: Lecture Notes in Computer Science, vol. 2719, Springer, 2003, pp. 384–396.
[8] S. Baswana, T. Kavitha, K. Mehlhorn, S. Pettie, New constructions of (α, β)-spanners and purely additive spanners, in: 16th Symposium on Discrete Algorithms, SODA, ACM–SIAM, 2005, pp. 672–681.
[9] C. Beeri, R. Fagin, D. Maier, M. Yannakakis, On the desirability of acyclic database schemes, J. ACM 30 (1983) 479–513.
[10] C. Berge, Hypergraphs, North Holland, 1989.
[11] P. Berman, A. Bhattacharyya, K. Makarychev, S. Raskhodnikova, G. Yaroslavtsev, Improved approximation for the directed spanner problem, in: Proc. ICALP'11, in: Lecture Notes in Computer Science, vol. 6755, Springer, Berlin, 2011, pp. 1–12.
[12] A. Bhattacharyya, E. Grigorescu, K. Jung, S. Raskhodnikova, D.P. Woodruff, Transitive-closure spanners, in: SODA 2009, 2009, pp. 932–941.
[13] U. Brandes, D. Handke, *NP*-completeness results for minimum planar spanners, Discrete Math. Theor. Comput. Sci. 3 (1998) 1–10.
[14] A. Brandstädt, V. Chepoi, F. Dragan, Distance approximating trees for chordal and dually chordal graphs, J. Algorithms 30 (1999) 166–184.
[15] A. Brandstädt, F.F. Dragan, H.-O. Le, V.B. Le, Tree spanners on chordal graphs: complexity and algorithms, Theoret. Comput. Sci. 310 (2004) 329–354.
[16] A. Brandstädt, F.F. Dragan, H.-O. Le, V.B. Le, R. Uehara, Tree spanners for bipartite graphs and probe interval graphs, Algorithmica 47 (2007) 27–51.
[17] L. Cai, D.G. Corneil, Tree spanners, SIAM J. Discrete Math. 8 (1995) 359–387.
[18] M. Charikar, C. Chekuri, A. Goel, S. Guha, S.A. Plotkin, Approximating a finite metric by a small number of tree metrics, in: FOCS, 1998, pp. 379–388.
[19] L.P. Chew, There are planar graphs almost as good as the complete graph, J. Comput. System Sci. 39 (1989) 205–219.
[20] D.G. Corneil, F.F. Dragan, E. Köhler, C. Yan, Collective tree 1-spanners for interval graphs, in: Proceedings of the 31st International Workshop "Graph-Theoretic Concepts in Computer Science", WG '05, June 2005, in: Lecture Notes in Computer Science, vol. 3787, Springer, 2005, pp. 151–162.
[21] R. Diestel, Graph Theory, second edition, Graduate Texts in Mathematics, vol. 173, Springer, 2000.
[22] M. Dinitz, R. Krauthgamer, Directed spanners via flow-based linear programs, in: STOC 2011, 2011, pp. 323–332.
[23] M. Dinitz, G. Kortsarz, R. Raz, Label Cover instances with large girth and the hardness of approximating basic k-spanner, CoRR abs, arXiv:1203.0224, 2012.
[24] D. Dor, S. Halperin, U. Zwick, All-pairs almost shortest paths, SIAM J. Comput. 29 (2000) 1740–1759.
[25] Y. Dourisboure, C. Gavoille, Tree-decompositions with bags of small diameter, Discrete Math. 307 (2007) 2008–2029.
[26] Y. Dourisboure, F.F. Dragan, C. Gavoille, C. Yan, Spanners for bounded tree-length graphs, Theoret. Comput. Sci. 383 (2007) 34–44.
[27] F.F. Dragan, M. Abu-Ata, Collective additive tree spanners of bounded tree-breadth graphs with generalizations and consequences, in: SOFSEM 2013: Theory and Practice of Computer Science, in: Lecture Notes in Computer Science, vol. 7741, 2013, pp. 194–206.
[28] F.F. Dragan, E. Köhler, An approximation algorithm for the tree t-spanner problem on unweighted graphs via generalized chordal graphs, in: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – Proceedings of the 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17–19, 2011, in: Lecture Notes in Computer Science, vol. 6845, Springer, 2011, pp. 171–183.
[29] F.F. Dragan, C. Yan, Collective tree spanners in graphs with bounded parameters, Algorithmica 57 (2010) 22–43.
[30] F.F. Dragan, C. Yan, D.G. Corneil, Collective tree spanners and routing in AT-free related graphs, J. Graph Algorithms Appl. 10 (2006) 97–122.

[31] F.F. Dragan, C. Yan, I. Lomonosov, Collective tree spanners of graphs, SIAM J. Discrete Math. 20 (2006) 241–260.
[32] F.F. Dragan, F.V. Fomin, P.A. Golovach, Spanners in sparse graphs, J. Comput. System Sci. 77 (2011) 1108–1119.
[33] F.F. Dragan, F.V. Fomin, P.A. Golovach, Approximation of minimum weight spanners for sparse graphs, Theoret. Comput. Sci. 412 (2011) 846–852.
[34] W. Duckworth, N.C. Wormald, M. Zito, A PTAS for the sparsest 2-spanner of 4-connected planar triangulations, J. Discrete Algorithms 1 (2003) 67–76.
[35] M. Elkin, D. Peleg, Strong inapproximability of the basic k-spanner problem, in: U. Montanari, J.D.P. Rolim, E. Welzl (Eds.), ICALP, in: Lecture Notes in Computer Science, vol. 1853, Springer, 2000, pp. 636–647.
[36] M. Elkin, D. Peleg, $(1+\epsilon, \beta)$-spanner constructions for general graphs, in: 33rd Annual ACM Symposium on Theory of Computing, STOC, Hersonissos, Crete, Greece, 2001, pp. 173–182.
[37] M. Elkin, D. Peleg, Approximating k-spanner problems for $k \geq 2$, Theoret. Comput. Sci. 337 (2005) 249–277.
[38] M. Elkin, D. Peleg, The hardness of approximating spanner problems, Theory Comput. Syst. 41 (2007) 691–729.
[39] M. Elkin, Y. Emek, D.A. Spielman, S.-H. Teng, Lower-stretch spanning trees, SIAM J. Comput. 38 (2008) 608–628.
[40] Y. Emek, D. Peleg, Approximating minimum max-stretch spanning trees on unweighted graphs, SIAM J. Comput. 38 (2008) 1761–1781.
[41] J. Fakcharoenphol, S. Rao, K. Talwar, A tight bound on approximating arbitrary metrics by tree metrics, J. Comput. System Sci. 69 (2004) 485–497.
[42] S.P. Fekete, J. Kremer, Tree spanners in planar graphs, Discrete Appl. Math. 108 (2001) 85–103.
[43] F.V. Fomin, P.A. Golovach, E. Jan van Leeuwen, Spanners of bounded degree graphs, Inform. Process. Lett. 111 (2011) 142–144.
[44] J.R. Gilbert, D.J. Rose, A. Edenbrandt, A separator theorem for chordal graphs, SIAM J. Algebr. Discrete Methods 5 (1984) 306–313.
[45] A. Gupta, A. Kumar, R. Rastogi, Traveling with a Pez dispenser (or, routing issues in MPLS), SIAM J. Comput. 34 (2004) 453–474.
[46] T.W. Haynes, S. Hedetniemi, P. Slater, Fundamentals of Domination in Graphs, Marcel Dekker, 1998.
[47] M. Herlihy, F. Kuhn, S. Tirthapura, R. Wattenhofer, Dynamic analysis of the arrow distributed protocol, Theory Comput. Syst. 39 (2006) 875–901.
[48] G. Kortsarz, On the hardness of approximating spanners, Algorithmica 30 (2001) 432–450.
[49] G. Kortsarz, D. Peleg, Generating sparse 2-spanners, J. Algorithms 17 (1994) 222–236.
[50] C. Liebchen, G. Wünsch, The zoo of tree spanner problems, Discrete Appl. Math. 156 (2008) 569–587.
[51] A.L. Liestman, T. Shermer, Additive graph spanners, Networks 23 (1993) 343–364.
[52] N. Linial, E. London, Y. Rabinovich, The geometry of graphs and some of its algorithmic applications, Combinatorica 15 (1995) 215–245.
[53] D. Lokshtanov, On the complexity of computing tree-length, Discrete Appl. Math. 158 (2010) 820–827.
[54] D. Peleg, Low stretch spanning trees, in: Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science, MFCS 2002, in: Lecture Notes in Computer Science, vol. 2420, 2002, pp. 68–80.
[55] D. Peleg, E. Reshef, Low complexity variants of the arrow distributed directory, J. Comput. System Sci. 63 (2001) 474–485.
[56] D. Peleg, A.A. Schäffer, Graph spanners, J. Graph Theory 13 (1989) 99–116.
[57] D. Peleg, J.D. Ullman, An optimal synchronizer for the hypercube, in: Proc. 6th ACM Symposium on Principles of Distributed Computing, Vancouver, 1987, 1987, pp. 77–85.
[58] D. Peleg, J.D. Ullman, An optimal synchronizer for the hypercube, SIAM J. Comput. 18 (1989) 740–747.
[59] D. Peleg, E. Upfal, A tradeoff between space and efficiency for routing tables (extended abstract), in: STOC, ACM, 1988, pp. 43–52.
[60] E. Prisner, Distance approximating spanning trees, in: Proc. STACS'97, in: Lecture Notes in Computer Science, vol. 1200, Springer, Berlin, 1997, pp. 499–510.
[61] E. Prisner, D. Kratsch, H.-O. Le, H. Müller, D. Wagner, Additive tree spanners, SIAM J. Discrete Math. 17 (2003) 332–340.
[62] N. Robertson, P.D. Seymour, Graph minors. II. Algorithmic aspects of tree-width, J. Algorithms 7 (1986) 309–322.
[63] M. Thorup, U. Zwick, Approximate distance oracles, J. ACM 52 (2005) 1–24.
[64] D.P. Woodruff, Additive spanners in nearly quadratic time, in: Proc. ICALP'10, in: Lecture Notes in Computer Science, vol. 6198, Springer, Berlin, 2010, pp. 463–474.
[65] C. Yan, Y. Xiang, F.F. Dragan, Compact and low delay routing labeling scheme for unit disk graphs, Comput. Geom. Theory Appl. 45 (2012) 305–325.