2006

Collective Tree Spanners of Graphs

Feodor F. Dragan
Kent State University - Kent Campus, fdragan@kent.edu

Chenyu Yan

Irina Lomonosov
Hiram College

SIAM J. DISCRETE MATH.
Vol. 20, No. 1, pp. 240–260

COLLECTIVE TREE SPANNERS OF GRAPHS*

FEODOR F. DRAGAN†, CHENYU YAN†, AND IRINA LOMONOSOV‡

Abstract. In this paper we introduce a new notion of *collective tree spanners*. 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 $d_T(x, y) \leq d_G(x, y) + r$. Among other results, we show that any chordal graph, chordal bipartite graph or cocomparability graph admits a system of at most $\log_2 n$ collective additive tree 2-spanners. These results are complemented by lower bounds, which say that any system of collective additive tree 1-spanners must have $\Omega(\sqrt{n})$ spanning trees for some chordal graphs and $\Omega(n)$ spanning trees for some chordal bipartite graphs and some cocomparability graphs. Furthermore, we show that any c-chordal graph admits a system of at most $\log_2 n$ collective additive tree $(2\lfloor c/2 \rfloor)$-spanners, any circular-arc graph admits a system of two collective additive tree 2-spanners. Towards establishing these results, we present a general property for graphs, called (α, r)-*decomposition*, and show that any (α, r)-decomposable graph G with n vertices admits a system of at most $\log_{1/\alpha} n$ collective additive tree $2r$-spanners. We discuss also an application of the collective tree spanners to the problem of designing compact and efficient routing schemes in graphs. For any graph on n vertices admitting a system of at most μ collective additive tree r-spanners, there is a routing scheme of deviation r with addresses and routing tables of size $O(\mu \log^2 n / \log \log n)$ bits per vertex. This leads, for example, to a routing scheme of deviation $(2\lfloor c/2 \rfloor)$ with addresses and routing tables of size $O(\log^3 n / \log \log n)$ bits per vertex on the class of c-chordal graphs.

Key words. sparse spanners, tree spanners, graph distance, balanced separator, graph decomposition, chordal graphs, c-chordal graphs, message routing, efficient algorithms

AMS subject classifications. 05C05, 05C10, 05C12, 05C78, 05C85, 94C15, 68R10, 68Q25, 68W25

DOI. 10.1137/S089548010444167X

1. Introduction.

Many combinatorial and algorithmic problems are concerned with the distance d_G on the vertices of a possibly weighted graph $G = (V, E)$. Approximating d_G by a simpler distance (in particular, by tree-distance d_T) is useful in many areas such as communication networks, data analysis, motion planning, image processing, network design, and phylogenetic analysis (see [1, 8, 11, 19, 22, 52, 58, 59, 64, 66]). An arbitrary metric space (in particular a finite metric defined by a general graph) might not have enough structure to exploit algorithmically; on trees, since they have a simpler (acyclic) structure, many hard algorithmic problems have easy solutions. So, the general goal is, for a given graph G, to find a simpler (well-structured, sparse, etc.) graph $H = (V, E')$ with the same vertex-set such that the distance $d_H(u, v)$ in H between two vertices $u, v \in V$ is reasonably close to the corresponding distance $d_G(u, v)$ in the original graph G.

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 is called a *multiplicative* t*-spanner* of G [22, 59, 58] if $d_H(u, v) \leq t \cdot d_G(u, v)$ for all $u, v \in V$. 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*

*Received by the editors March 5, 2004; accepted for publication (in revised form) November 1, 2005; published electronically March 15, 2006. Results of this paper were partially presented at the SWAT '04 conference [30].

http://www.siam.org/journals/sidma/20-1/44167.html

†Department of Computer Science, Kent State University, Kent, OH 44242 (dragan@cs.kent.edu, cyan@cs.kent.edu).

‡Department of Computer Science, Hiram College, Hiram, OH 44234 (lomonosovi@hiram.edu).

of G [52]. The parameters t and r are called, respectively, the *multiplicative* and the *additive stretch factors*. Clearly, every additive r-spanner of G is a multiplicative $(r+1)$-spanner of G (but not vice versa). Note that the graphs considered in this paper are assumed to be unweighted (except in section 7 where we discuss how to extend our results to weighted graphs).

Graph spanners have applications in various areas, especially in distributed systems and communication networks. In [59], close relationships were established between the quality of spanners (in terms of stretch factor and the number of spanner edges $|E'|$), and the time and communication complexities of any synchronizer for the network based on this spanner. Also, 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 [60]. Unfortunately, the problem of determining, for a given graph G and two integers $t \geq 2, m \geq 1$, whether G has a multiplicative t-spanner with m or fewer edges, is NP-complete (see [58]).

The sparsest spanners are tree spanners. Tree spanners occur in biology [5], and as it was shown in [57], they can be used as models for broadcast operations in communication networks. Tree spanners are favored also from the algorithmic point of view—many algorithmic problems are easily solvable on trees. Multiplicative tree t-spanners were studied in [19]. It was shown that, for a given graph G, the problem to decide whether G has a multiplicative tree t-spanner (the *multiplicative tree t-spanner problem*) is NP-complete for any fixed $t \geq 4$ and is linearly solvable for $t = 1, 2$. Recently, this NP-completeness result was improved—the multiplicative tree t-spanner problem is NP-complete for any fixed $t \geq 4$ even on some rather restricted graph classes: planar graphs [12], chordal graphs [14] and chordal bipartite graphs [15].

Nevertheless, some particular graph classes, such as cographs, complements of bipartite graphs, split graphs, regular bipartite graphs, interval graphs, permutation graphs, convex bipartite graphs, distance-hereditary graphs, directed path graphs, cocomparability graphs, AT-free graphs, strongly chordal graphs, and dually chordal graphs do admit additive tree r-spanners and/or multiplicative tree t-spanners for sufficiently small r and t (see [13, 18, 51, 55, 61, 62, 69]). We refer also to [1, 12, 14, 18, 19, 38, 52, 57, 58, 65] for more background information on tree and general sparse spanners.

Many graph classes (including hypercubes, planar graphs, chordal graphs, chordal bipartite graphs) do not admit any good tree spanner. For every fixed integer t there are planar chordal graphs and planar chordal bipartite graphs that do not admit tree t-spanners (additive as well as multiplicative) [21, 62]. However, as it was shown in [58], any chordal graph with n vertices admits a multiplicative 5-spanner with at most $2n-2$ edges and a multiplicative 3-spanner with at most $O(n \log n)$ edges (both spanners are constructable in polynomial time). Recently, the results were further improved. In [21], the authors show that every chordal graph admits an additive 4-spanner with at most $2n-2$ edges and an additive 3-spanner with at most $O(n \log n)$ edges. An additive 4-spanner can be constructed in linear time while an additive 3-spanner is constructable in $O(m \log n)$ time, where m is the number of edges of G. Even more, the method designed for chordal graph is extended to all c-chordal graphs. As a result, it was shown that any such graph admits an additive $(c+1)$-spanner with at most $2n-2$ edges which is constructable in $O(cn+m)$ time. Recall that a graph G is *chordal* if its largest induced (chordless) cycles are of length 3 and *c-chordal* if its largest induced cycles are of length c. Note also that [59] gives a method for constructing a multiplicative 3-spanner of the n-vertex hypercube with fewer than $7n$

edges and this construction was improved in [34] to give a multiplicative 3-spanner of the n-vertex hypercube with fewer than $4n$ edges.

1.1. Our results. In this paper we introduce a new notion of *collective tree spanners*, a notion 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. Furthermore, any result on collective *additive* tree spanners can be translated into a result on collective *multiplicative* tree spanners since any graph, admitting a system of μ collective *additive* tree r-spanners, admits a system of μ collective *multiplicative* tree $(r + 1)$-spanners ($d_T(x, y) \leq d_G(x, y) + r$ implies $d_T(x, y)/d_G(x, y) \leq 1 + r/d_G(x, y) \leq r + 1$ for an unweighted graph G). Note also that any graph on n vertices admits a system of at most $n - 1$ collective additive tree 0-spanners (take $n - 1$ breadth-first-search–trees rooted at different vertices of G).

The introduction of this new notion was inspired by the works [6, 7] of Bartal and subsequent works [20, 37]. For example, motivated by Bartal's work on probabilistic approximation of general metrics with tree metrics, [20] gives a polynomial time algorithm that given a finite n point metric G, constructs $O(n \log n)$ trees and a probability distribution ψ on them such that the expected multiplicative stretch of any edge of G in a tree chosen according to ψ is at most $O(\log n \log \log n)$. These results led to approximation algorithms for a number of optimization problems including the group Steiner tree problem, the metric labeling problem, the buy-at-bulk network design problem and many others (see [6, 7, 20, 37] for more details).

In section 2 we define a large class of graphs, called (α, r)-*decomposable*, and show that any (α, r)-decomposable graph G with n vertices admits a system of at most $\log_{1/\alpha} n$ collective additive tree $2r$-spanners. Then, in sections 3 and 4, we show that chordal graphs, chordal bipartite graphs, and cocomparability graphs are all $(1/2, 1)$-decomposable graphs, implying that each graph from those families admits a system of at most $\log_2 n$ collective additive tree 2-spanners. These results are complemented by lower bounds, which say that any system of collective additive tree 1-spanners must have $\Omega(\sqrt{n})$ spanning trees for some chordal graphs and $\Omega(n)$ spanning trees for some chordal bipartite graphs and some cocomparability graphs. Furthermore, we show that any c-chordal graph is $(1/2, \lfloor c/2 \rfloor)$-decomposable, implying that each c-chordal graph admits a system of at most $\log_2 n$ collective additive tree $(2\lfloor c/2 \rfloor)$-spanners.

Thus, as a byproduct, we get that chordal graphs, chordal bipartite graphs, and cocomparability graphs admit additive 2-spanners with at most $(n - 1) \log_2 n$ edges and c-chordal graphs admit additive $(2\lfloor c/2 \rfloor)$-spanners with at most $(n - 1) \log_2 n$ edges. Our result for chordal graphs improves the known results from [58] and [21] on 3-spanners and answers the question posed in [21] whether chordal graphs admit additive 2-spanners with $O(n \log n)$ edges.

In section 5, we show that each circular-arc graph admits a system of two collective additive tree 2-spanners, and that for any constant $r \geq 0$ there is a circular-arc graph without any (one) additive tree r-spanner.

In section 6 we discuss an application of the collective tree spanners to the problem of designing compact and efficient routing schemes in graphs. For any graph

on n vertices admitting a system of at most μ collective additive tree r-spanners, there is a routing scheme of deviation r with addresses and routing tables of size $O(\mu \log^2 n / \log\log n)$ bits per vertex (for details see section 6). This leads, for example, to a routing scheme of deviation $(2\lfloor c/2 \rfloor)$ with addresses and routing tables of size $O(\log^3 n / \log\log n)$ bits per vertex on the class of c-chordal graphs. The latter improves the recent result on routing on c-chordal graphs obtained in [33] (see also [32] for the case of chordal graphs). We conclude the paper with section 7, where we discuss how to extend our results to weighted graphs, and section 8, where we discuss some further developments and future directions.

1.2. Basic notions and notations. All graphs occurring in this paper are connected, finite, undirected, loopless and without multiple edges. In a graph $G = (V, E)$ 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 the vertices u and v is the length of a shortest path connecting u and v.

For a subset $S \subseteq V$, let $rad_G(S)$ and $diam_G(S)$ be the radius and the diameter, respectively, of S in G, i.e., $rad_G(S) = min_{v \in V}\{max_{u \in S}\{d_G(u, v)\}\}$, $diam_G(S) = max_{u,v \in S}\{d_G(u, v)\}$. A vertex $v \in V$ such that $d_G(u, v) \leq rad_G(S)$ for any $u \in S$ is called a central vertex for S. The value $rad_G(V)$ is called *the radius* of G. Let also $N(v)$ ($N[v]$) denote the open (closed) neighborhood of a vertex v in G, i.e., $N(v) = \{u \in V : uv \in E(G)\}$ and $N[v] = N(v) \cup \{v\}$.

2. (α, r)-decomposable graphs and their collective tree spanners. Different balanced separators in graphs were used by many authors in designing efficient graph algorithms (see [26, 27, 43, 44, 46, 50, 53, 54]). For example, bounded size balanced separators and bounded diameter balanced separators were recently employed in [43, 44, 50] for designing compact distance labeling schemes for different so-called well-separated families of graphs. We extend those ideas and apply them to our problem.

Let α be a positive real number smaller than 1 and r be a nonnegative integer. We say that an n-vertex graph $G = (V, E)$ is (α, r)-*decomposable* if there is a separator $S \subseteq V$ such that the following three conditions hold:

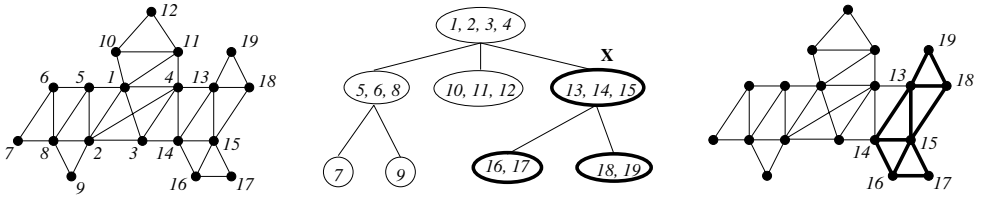
balanced separator condition—the removal of S leaves no connected component with more than αn vertices;

bounded separator-radius condition—$rad_G(S) \leq r$, i.e., there exists a vertex c in G (called *a central vertex for* S) such that $d_G(v, c) \leq r$ for any $v \in S$;

hereditary family condition—each connected component of the graph, obtained from G by removing vertices of S, is also an (α, r)-decomposable graph.

Note that, by definition, any graph of radius at most r is (α, r)-decomposable and that the size of S does not matter.

2.1. Collective tree spanners of (α, r)-decomposable graphs. Using the first and third conditions of the definition, one can construct for any (α, r)-decomposable graph G a *(rooted) balanced decomposition tree* $\mathcal{BT}(G)$ as follows. If G is of radius at most r, then $\mathcal{BT}(G)$ is a one-node tree. Otherwise, find a balanced separator S in G, which exists according to the balanced separator condition. Let $G_1, G_2, \ldots, G_p$ be the connected components of the graph $G - S$ obtained from G by removing vertices of S. For each graph G_i ($i = 1, \ldots, p$), which is (α, r)-decomposable by the hereditary family condition, construct a balanced decomposition tree $\mathcal{BT}(G_i)$ recursively, and build $\mathcal{BT}(G)$ by taking S to be the root and connecting the root of each tree $\mathcal{BT}(G_i)$ as a child of S. See Figure 1 for an illustration. Clearly, the nodes of $\mathcal{BT}(G)$ represent a partition of the vertex set V of G into *clusters* $S_1, S_2, \ldots, S_q$ of radius at most r



(a) (b) (c)

FIG. 1. (a) *A graph* G, (b) *its balanced decomposition tree* $\mathcal{BT}(G)$, *and* (c) *an induced subgraph* $G(\downarrow X)$ *of* G.

each. For a node X of $\mathcal{BT}(G)$, denote by $G(\downarrow X)$ the (connected) subgraph of G induced by vertices $\bigcup\{Y : Y$ is a descendent of X in $\mathcal{BT}(G)\}$ (here we assume that X is a descendent of itself).

It is easy to see that a balanced decomposition tree $\mathcal{BT}(G)$ of a graph G with n vertices and m edges has depth at most $\log_{1/\alpha} n$, which is $O(log_2 n)$ if α is a constant. Moreover, assuming that a balanced and bounded radius separator can be found in polynomial, say $p(n)$, time (for the special graph classes we consider later, $p(n)$ will be at most $O(n^3)$), the tree $\mathcal{BT}(G)$ can be constructed in $O((p(n)+m)\log_{1/\alpha} n)$ total time. Indeed, in each level of recursion we need to find balanced and bounded radius separators in current disjoint subgraphs and to construct the corresponding subgraphs of the next level. Also, since the graph sizes are reduced by a factor α, the recursion depth is at most $\log_{1/\alpha} n$.

Consider now two arbitrary vertices x and y of an (α, r)-decomposable graph G and let $S(x)$ and $S(y)$ be the nodes of $\mathcal{BT}(G)$ containing x and y, respectively. Let also $NCA_{\mathcal{BT}(G)}(S(x), S(y))$ be the nearest common ancestor of nodes $S(x)$ and $S(y)$ in $\mathcal{BT}(G)$ and $(X_0, X_1, \ldots, X_t)$ be the path of $\mathcal{BT}(G)$ connecting the root X_0 of $\mathcal{BT}(G)$ with $NCA_{\mathcal{BT}(G)}(S(x), S(y)) = X_t$ (in other words, $X_0, X_1, \ldots, X_t$ are the common ancestors of $S(x)$ and $S(y)$). The following lemmata are crucial to all our subsequent results.

LEMMA 2.1. *Any path* $P^G_{x,y}$, *connecting vertices* x *and* y *in* G, *contains a vertex from* $X_0 \cup X_1 \cup \cdots \cup X_t$.

Let $SP^G_{x,y}$ be a shortest path of G connecting vertices x and y, and let X_i be the node of the path $(X_0, X_1, \ldots, X_t)$ with the smallest index such that $SP^G_{x,y} \bigcap X_i \neq \emptyset$ in G. Then, the following lemma holds.

LEMMA 2.2. *We have* $d_G(x, y) = d_{G'}(x, y)$, *where* $G' := G(\downarrow X_i)$.

Proof. It is enough to show that the path $SP^G_{x,y}$ consists of only vertices of G'. Let us assume, by 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 $\mathcal{BT}(G)$, containing vertex z, is not a descendent of X_i. Therefore, the nearest common ancestor of $S(x)$ and $S(z)$ in $\mathcal{BT}(G)$ is a node X_j from $\{X_0, X_1, \ldots, X_t\}$ with $j < i$. But then, by Lemma 2.1, the path $SP^G_{x,z}$ (and hence the path $SP^G_{x,y}$) must have a vertex in $X_0 \cup X_1 \cup \cdots \cup X_j$, contradicting the choice of X_i, $i > j$. □

For the graph $G' = G(\downarrow X_i)$, consider its arbitrary *breadth-first-search–tree (BFS-tree)* T' rooted at a central vertex c for X_i, i.e., a vertex c such that $d_{G'}(v, c) \leq r$ for any $v \in X_i$. Such a vertex exists in G' since G' is an (α, r)-decomposable graph and X_i is its balanced and bounded radius separator. The tree T' has the following distance property with respect to those vertices x and y.

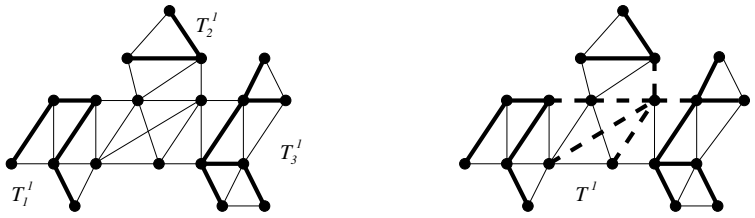


FIG. 2. (a) *Local subtrees* T_1^1, T_2^1, T_3^1 *of graph* G *from Figure* 1 *and* (b) *a corresponding spanning tree* T^1 *of* G *(dark solid edges are edges of local subtrees* T_1^1, T_2^1, T_3^1*, dashed edges are added to create one spanning tree* T^1 *on top of* T_1^1, T_2^1, T_3^1*).*

LEMMA 2.3. *We have* $d_{T'}(x,y) \le d_G(x,y) + 2r$.

Proof. We know, by Lemma 2.2, that a shortest path $SP^G_{x,y}$, intersecting X_i and not intersecting any X_l $(l < i)$, lies entirely in G'. Let x' be the vertex of $SP^G_{x,y} \cap X_i$ closest to x and y' be the vertex of $SP^G_{x,y} \cap X_i$ closest to y. Since T' is a BFS-tree of G' rooted at vertex c, we have

$$d_{T'}(x,c) = d_{G'}(x,c) \le d_{G'}(x,x') + d_{G'}(x',c) \le d_{G'}(x,x') + r = d_G(x,x') + r,$$
$$d_{T'}(y,c) = d_{G'}(y,c) \le d_{G'}(y,y') + d_{G'}(y',c) \le d_{G'}(y,y') + r = d_G(y,y') + r.$$

That is, $d_{T'}(x,y) \le d_{T'}(x,c) + d_{T'}(y,c) \le d_G(x,x') + d_G(y,y') + 2r$. Combining this with the fact that $d_G(x,y) \ge d_G(x,x') + d_G(y,y')$, we obtain $d_{T'}(x,y) \le d_G(x,y) + 2r$. □

Let now $B_1^i, \ldots, B_{p_i}^i$ be the nodes on depth i of the tree $\mathcal{BT}(G)$. For each subgraph $G_j^i := G(\downarrow B_j^i)$ of G $(i = 0, 1, \ldots, depth(\mathcal{BT}(G)),\ j = 1, 2, \ldots, p_i)$, denote by T_j^i a BFS-tree of graph G_j^i rooted at a central vertex c_j^i for B_j^i (see Figure 2 for an illustration). The trees T_j^i $(i = 0, 1, \ldots, depth(\mathcal{BT}(G)),\ j = 1, 2, \ldots, p_i)$ are called *local subtrees* of G, and, given the balanced decomposition tree $\mathcal{BT}(G)$, they can be constructed in $O((t(n)+m)\log_{1/\alpha} n)$ total time, where $t(n)$ is the time needed to find a central vertex c_j^i for B_j^i (a trivial upper bound for $t(n)$ is $O(n^3)$). From Lemma 2.3 the following general result can be deduced.

THEOREM 2.4. *Let* G *be an* (α, r)*-decomposable graph,* $\mathcal{BT}(G)$ *be its balanced decomposition tree and* $\mathcal{LT}(G) = \{T_j^i : i = 0, 1, \ldots, depth(\mathcal{BT}(G)),\ j = 1, 2, \ldots, p_i\}$ *be its local subtrees. Then, for any two vertices* x *and* y *of* G*, there exists a local subtree* $T_{j'}^{i'}$ *in* $\mathcal{LT}(G)$ *such that*

$$d_{T_{j'}^{i'}}(x,y) \le d_G(x,y) + 2r.$$

This theorem implies two important results for the class of (α, r)-decomposable graphs. Let G be an (α, r)-decomposable graph with n vertices and m edges, $\mathcal{BT}(G)$ be its balanced decomposition tree, and $\mathcal{LT}(G)$ be the family of 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.,

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

Clearly, H is a spanning subgraph of G, constructable in $O((p(n)+t(n)+m)\log_{1/\alpha} n)$ total time, and, for any two vertices x and y of G, $d_H(x,y) \le d_G(x,y)+2r$ holds. Also,

since for every level i ($i = 0, 1, \ldots, depth(\mathcal{BT}(G))$) of balanced decomposition tree $\mathcal{BT}(G)$, 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 $(n-1)\log_{1/\alpha} n$ edges in total. Thus, we have proven the following result.

THEOREM 2.5. *Any (α, r)-decomposable graph G with n vertices admits an additive $2r$-spanner with at most $(n-1)\log_{1/\alpha} n$ edges.*

Instead of taking the union of all local subtrees of G, one can fix i ($i \in \{0, 1, \ldots, depth(\mathcal{BT}(G))\}$) and consider separately the union of only local subtrees $T_1^i, \ldots, T_{p_i}^i$, corresponding to the level i of the decomposition tree $\mathcal{BT}(G)$, 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 balanced decomposition* $\mathcal{BT}(G)$. In this way we can obtain at most $\log_{1/\alpha} n$ spanning trees for G, one for each level i of $\mathcal{BT}(G)$. Denote the collection of those spanning trees by $\mathcal{T}(G)$. By Theorem 2.4, it is rather straightforward to show that for any two vertices x and y of G, there exists a spanning tree $T^{i'}$ in $\mathcal{T}(G)$ such that $d_{T^{i'}}(x, y) \leq d_G(x, y) + 2r$. Thus, we have the following theorem.

THEOREM 2.6. *Any (α, r)-decomposable graph G with n vertices admits a system $\mathcal{T}(G)$ of at most $\log_{1/\alpha} n$ collective additive tree $2r$-spanners.*

Note that such a system $\mathcal{T}(G)$ for an (α, r)-decomposable graph G with n vertices and m edges can be constructed in $O((p(n) + t(n) + m)\log_{1/\alpha} n)$ time, where $p(n)$ is the time needed to find a balanced and bounded radius separator S and $t(n)$ is the time needed to find a central vertex for S.

2.2. Extracting an appropriate tree from $\mathcal{T}(G)$. Now we will show that one can assign $O(\log_{1/\alpha} n \times \log n)$ bit labels to vertices of G such that, for any pair of vertices x and y, a tree $T^{i'}$ in $\mathcal{T}(G)$ with $d_{T^{i'}}(x, y) \leq d_G(x, y) + 2r$ can be identified in only $O(\log_{1/\alpha} n)$ time by merely inspecting the labels of x and y, without using any other information about the graph. This will be useful in an application of collective tree spanners, discussed in section 6.

Associate with each vertex x of G a $2 \times (depth(\mathcal{BT}(G)) + 1)$ array A_x such that, for each level i of $\mathcal{BT}(G)$, $A_x[1, i] = j$ and $A_x[2, i] = d_{T_j^i}(x, c_j^i)$ if there exists a local subtree T_j^i in $\mathcal{LT}(G)$ containing vertex x, and $A_x[1, i] = nil$ and $A_x[2, i] = \infty$, otherwise (i.e., the depth in $\mathcal{BT}(G)$ of node $S(x)$ containing x is smaller than i). Evidently, each label A_x ($x \in V$) can be encoded using $O(\log_{1/\alpha} n \times \log n)$ bits and a computation of all labels A_x, $x \in V$ can be performed together with the construction of system $\mathcal{T}(G)$.

Given labels A_x, A_y of vertices x and y, the following procedure will return in $O(\log_{1/\alpha} n)$ time an index $i' \in \{0, 1, \ldots, depth(\mathcal{BT}(G))\}$ such that, for tree $T^{i'} \in \mathcal{T}(G)$, $d_{T^{i'}}(x, y) \leq d_G(x, y) + 2r$ holds.

set $i' := 0$;
set $minsum := A_x[2, 0] + A_y[2, 0]$;
set $i := 1$;
while $(A_x[1, i] = A_y[1, i] \neq nil)$ and $(i \leq \log_{1/\alpha} n)$ do
  if $A_x[2, i] + A_y[2, i] < minsum$
    then set $i' := i$ and $minsum := A_x[2, i] + A_y[2, i]$;
  $i := i + 1$;
enddo
return i'.

This procedure simply finds, among all local subtrees containing both x and y, a subtree $T^{i'}_{j'}$ for which the sum $d_{T^{i'}_{j'}}(x, c^{i'}_{j'}) + d_{T^{i'}_{j'}}(y, c^{i'}_{j'})$ is minimum, and then returns its upper index i'.

To show that indeed $d_{T^{i'}}(x, y) \leq d_G(x, y) + 2r$, we will need to recall the proof of Lemma 2.3 (note that $d_{T^{i'}}(x, y) = d_{T^{i'}_{j'}}(x, y)$ by construction of $T^{i'}$). Let again $S(x)$ and $S(y)$ be the nodes of $\mathcal{BT}(G)$ containing vertices x and y, respectively, and let $(B^0, B^1_{j_1}, \ldots, B^t_{j_t})$ be the path of $\mathcal{BT}(G)$ connecting the root B^0 of $\mathcal{BT}(G)$ with $NCA_{\mathcal{BT}(G)}(S(x), S(y)) = B^t_{j_t}$. In Lemma 2.3 we proved that there exists an index $i \in \{0, 1, \ldots, t\}$ such that any BFS-tree T' of the graph $G(\downarrow B^i_{j_i})$ rooted at a center c for $B^i_{j_i}$ (including local subtree $T^i_{j_i}$ rooted at $c^i_{j_i}$) satisfies $d_{T'}(x, y) \leq d_{T'}(x, c) + d_{T'}(y, c) \leq d_G(x, y) + 2r$ (see inequalities (1) and (2) in that proof). Since, among local subtrees $T^0, T^1_{j_1}, \ldots, T^t_{j_t}$, the subtree $T^{i'}_{j'}$ has minimum sum $d_{T^{i'}_{j'}}(x, c^{i'}_{j'}) + d_{T^{i'}_{j'}}(y, c^{i'}_{j'})$, we conclude

$$\begin{aligned} d_{T^{i'}}(x, y) = d_{T^{i'}_{j'}}(x, y) &\leq d_{T^{i'}_{j'}}(x, c^{i'}_{j'}) + d_{T^{i'}_{j'}}(y, c^{i'}_{j'}) \\ &\leq d_{T^{i}_{j_i}}(x, c^{i}_{j_i}) + d_{T^{i}_{j_i}}(y, c^{i}_{j_i}) \leq d_G(x, y) + 2r. \end{aligned}$$

3. Acyclic hypergraphs, chordal graphs and (α, r)-decomposition. Let $H = (V, \mathcal{E})$ be a *hypergraph* with the vertex set V and the *hyperedge* set $\mathcal{E}$, i.e., $\mathcal{E}$ is a set of nonempty subsets of V. 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 (a set of pairwise adjacent vertices) 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. For these and other hypergraph notions see [10].

The following theorem represents two well-known characterizations of acyclic hypergraphs. Let $\mathcal{C}(G)$ be the set of all maximal (by inclusion) cliques of a graph $G = (V, E)$. The hypergraph $(V, \mathcal{C}(G))$ is called the clique-hypergraph of G. Recall that a graph G is *chordal* if it does not contain any induced cycles of length greater than 3.

THEOREM 3.1 (see [2, 9, 10, 17, 36, 67]). *Let $H = (V, \mathcal{E})$ be a hypergraph. Then the following conditions are equivalent:*

(i) *H is an acyclic hypergraph;*
(ii) *H is conformal and $2SEC(H)$ of H is a chordal graph;*
(iii) *H is the clique hypergraph $(V, \mathcal{C}(G))$ of some chordal graph $G = (V, E)$.*

Later we will need also the following known result. A vertex v of a graph G is called *simplicial* if its neighborhood $N(v)$ forms a clique in G.

THEOREM 3.2 (see [17, 25]). *Let $G = (V, E)$ be a graph. Then the following conditions are equivalent:*

(i) *G is a chordal graph;*
(ii) *the clique hypergraph $(V, \mathcal{C}(G))$ of G is acyclic (in other words, G is the intersection graph of a family of subtrees of a tree);*
(iii) *G has a perfect elimination ordering. i.e., an ordering $v_1, v_2, \ldots, v_n$ of vertices of G such that, for any i, $i \in \{1, 2, \ldots, n\}$, vertex v_i is simplicial in graph $G(v_i, \ldots, v_n)$, the subgraph of G induced by vertices $v_i, \ldots, v_n$.*

Let now $G = (V, E)$ be an arbitrary graph and r be a positive integer. We say that G *admits a radius r acyclic covering* if there is a family $\mathcal{S}(G) = \{S_1, \ldots, S_k\}$ of

subsets of V such that

(1) $\bigcup_{i=1}^{k} S_i = V$;
(2) for any edge xy of G there is a subset S_i ($i \in \{1, \ldots, k\}$) with $x, y \in S_i$;
(3) $H = (V, \mathcal{S}(G))$ is an acyclic hypergraph;
(4) $rad_G(S_i) \leq r$ for each $i = 1, \ldots, k$.

A class of graphs $\mathcal{F}$ is called *hereditary* if every induced subgraph of a graph G belongs to $\mathcal{F}$ whenever G is in $\mathcal{F}$. A class of graphs $\mathcal{F}$ is called (α, r)-*decomposable* if every graph G from $\mathcal{F}$ is (α, r)-decomposable.

THEOREM 3.3. *Let $\mathcal{F}$ be a hereditary class of graphs such that any $G \in \mathcal{F}$ admits a radius r acyclic covering. Then $\mathcal{F}$ is a $(1/2, r)$-decomposable class of graphs.*

Proof. Consider a graph $G \in \mathcal{F}$ and let $\mathcal{S}(G) = \{S_1, \ldots, S_k\}$ be its radius r acyclic covering. Since $H = (V, \mathcal{S}(G))$ is an acyclic hypergraph, $2SEC(H)$ is chordal and H is conformal. It is well known [47], that every n-vertex chordal graph Γ contains a maximal clique C such that if the vertices in C are deleted from Γ, every connected component in the graph induced by any remaining vertices is of size at most $n/2$. Moreover, according to [47], for any chordal graph on n vertices and m edges, such a separating clique C can be found in $O(n+m)$ time. Applying this result to an n-vertex chordal graph $2SEC(H)$, we will get in at most $O(n^2)$ time a maximal clique S of $2SEC(H)$ such that any connected component of the graph $2SEC(H) - S$ (obtained from $2SEC(H)$ by deleting vertices of S) has at most $n/2$ vertices. Since $2SEC(H)$ is obtained from G by adding some new edges, removing vertices of S from the original graph G will leave no connected component (in $G - S$) with more than $n/2$ vertices. Furthermore, since $\mathcal{F}$ is a hereditary class of graphs, all connected components of $G - S$ induce graphs from $\mathcal{F}$ (and they can be assumed by induction to be $(1/2, r)$-decomposable graphs). It remains to note that, from conformality of H, there must exist a set S_i in $\mathcal{S}(G)$ which contains S, that is, $rad_G(S) \leq rad_G(S_i) \leq r$ must hold. □

Since for a chordal graph $G = (V, E)$ the clique hypergraph $(V, \mathcal{C}(G))$ is acyclic and chordal graphs form a hereditary class of graphs, from Theorem 3.3 and Theorems 2.5 and 2.6, we immediately conclude the following corollaries.

COROLLARY 3.4. *Any chordal graph G with n vertices and m edges admits an additive 2-spanner with at most $(n-1)\log_2 n$ edges, and such a sparse spanner can be constructed in $O(m \log_2 n)$ time.*

COROLLARY 3.5. *Any chordal graph G with n vertices and m edges admits a system $\mathcal{T}(G)$ of at most $\log_2 n$ collective additive tree 2-spanners, and such a system of spanning trees can be constructed in $O(m \log_2 n)$ time.*

Note that, since any additive r-spanner is a multiplicative $(r+1)$-spanner, Corollary 3.4 improves a known result of Peleg and Schäffer on sparse spanners of chordal graphs. In [58], they proved that any chordal graph with n vertices admits a multiplicative 3-spanner with at most $O(n \log_2 n)$ edges and a multiplicative 5-spanner with at most $2n - 2$ edges. Both spanners can be constructed in polynomial time. Note also that their result on multiplicative 5-spanners was earlier improved in [21], where the authors showed that any chordal graph with n vertices admits an additive 4-spanner with at most $2n - 2$ edges, constructable in linear time. Motivated by this and Corollary 3.5, it is natural to ask whether a system of constant number of collective additive tree 4-spanners exists for a chordal graph (or, generally, for which r, a system of constant number of collective additive tree r-spanners exists for any chordal graph). Recall that the problem whether a chordal graph admits a (one) multiplicative tree t-spanner is NP-complete for any $t > 3$ [14].

Peleg and Schäffer showed also in [58] that there are n-vertex chordal graphs for which any multiplicative 2-spanner will need to have at least $\Omega(n^{3/2})$ edges. This result leads to the following observation on collective additive tree 1-spanners of chordal graphs.

OBSERVATION 3.6. *There are n-vertex chordal graphs for which any system of collective additive tree 1-spanners will need to have at least $\Omega(\sqrt{n})$ spanning trees.*

Proof. Indeed, the existence of a system of $o(\sqrt{n})$ collective additive tree 1-spanners for a chordal graph will lead to the existence of an additive 1-spanner (and hence, of a multiplicative 2-spanner) with $o(n^{3/2})$ edges. □

4. Collective tree spanners in c-chordal graphs. A graph G is *c-chordal* if it does not contain any induced cycles of length greater than c; c-chordal graphs naturally generalize the class of chordal graphs. Chordal graphs are precisely the 3-chordal graphs.

THEOREM 4.1. *The class of c-chordal graphs is $(1/2, \lfloor c/2 \rfloor)$-decomposable.*

Proof. By Theorem 3.3 and since c-chordal graphs form a hereditary class of graphs, we need only to show that any c-chordal graph G admits a radius $\lfloor c/2 \rfloor$ acyclic covering. The existence of a radius $\lfloor c/2 \rfloor$ acyclic covering for G easily follows from a famous result of [43], which states that any c-chordal graph $G = (V, E)$ admits a special kind of *Robertson and Seymour tree-decomposition* [63]. That is, a tree $\mathcal{DT}(G)$, whose nodes are subsets of V, exists such that

- $(1')$ $\bigcup\{S : S \text{ is a node of } \mathcal{DT}(G)\} = V$;
- $(2')$ for any edge xy of G there is a node S of $\mathcal{DT}(G)$ with $x, y \in S$;
- $(3')$ for any tree nodes X, Y, Z of $\mathcal{DT}(G)$, if Y is on the path from X to Z in $\mathcal{DT}(G)$, then $X \cap Z \subseteq Y$;
- $(4')$ $diam_G(S) \leq \lfloor c/2 \rfloor$ for each node S of $\mathcal{DT}(G)$.

The reader might notice a close similarity between these four properties and the four properties from the definition of a radius r acyclic covering. In fact, they are almost equivalent. Note that $diam_G(S) \leq \lfloor c/2 \rfloor$ implies $rad_G(S) \leq \lfloor c/2 \rfloor$. Let $\mathcal{S}(G) = \{S : S \text{ is a node of } \mathcal{DT}(G)\}$ and consider a hypergraph $H = (V, \mathcal{S}(G))$. We claim that for a family $\mathcal{S}(G)$ of subsets of V, properties (1), (2) and (3) are equivalent to properties $(1')$, $(2')$ and $(3')$. Indeed, since, by property $(3')$, $v \in X \cap Z$ implies v belongs to any Y on the path of $\mathcal{DT}(G)$ from X to Z, for any vertex $v \in V$ the elements of $\mathcal{S}(G)$ containing vertex v induce a subtree in $\mathcal{DT}(G)$. Hence, by definition, $H = (V, \mathcal{S}(G))$ is an acyclic hypergraph. Conversely, let that for a graph G, a family $\mathcal{S}(G)$ of subsets of V satisfies properties (1), (2) and (3). Then, the acyclicity of the hypergraph $H = (V, \mathcal{S}(G))$ implies the existence of a tree T with node set $\mathcal{S}(G)$ such that for any vertex $v \in V$, the elements of $\mathcal{S}(G)$ containing v induce a subtree in T. Therefore, if two nodes X and Z of the tree T contain a vertex v then any node Y of T between X and Z must contain v, too. □

A balanced separator of radius at most $\lfloor c/2 \rfloor$ of a c-chordal graph G on n vertices and m edges can be found in $O(n^3)$ time as follows. Use an $O(nm)$ time algorithm from [33] to construct a Robertson–Seymour tree-decomposition $\mathcal{DT}(G)$ of G (it will have at most n nodes [33]). Then define the family $\mathcal{S}(G) = \{S : S \text{ is a node of } \mathcal{DT}(G)\}$ and consider the 2-section graph $2SEC(H)$ of an acyclic hypergraph $H = (V, \mathcal{S}(G))$. $2SEC(H)$ can be constructed in at most $O(n^3)$ time. Using an algorithm from [47], find a balanced separator C of a chordal graph $2SEC(H)$ in $O(n^2)$ time. We know that C is a maximal clique of $2SEC(H)$ and there must exist a set $S \in \mathcal{S}(G)$ which coincides with C (by conformality of H). As we showed earlier (see the proof of Theorem 3.3), $C = S$ is a balanced separator of radius at most $\lfloor c/2 \rfloor$ of G.

Thus, from Theorems 2.5 and 2.6, we conclude the following corollaries.

COROLLARY 4.2. *Any c-chordal graph G with n vertices admits an additive $(2\lfloor c/2\rfloor)$-spanner with at most $(n-1)\log_2 n$ edges, and such a sparse spanner can be constructed in $O(n^3 \log_2 n)$ time.*

COROLLARY 4.3. *Any c-chordal graph G with n vertices admits a system $\mathcal{T}(G)$ of at most $\log_2 n$ collective additive tree $(2\lfloor c/2\rfloor)$-spanners, and such a system of spanning trees can be constructed in $O(n^3 \log_2 n)$ time.*

Note that there are c-chordal graphs which do not admit any radius r acyclic covering with $r < \lfloor c/2\rfloor$. Consider, for example, the complement $\overline{C_6}$ of an induced cycle $C_6 = (a-b-c-d-e-f-a)$, which is a 4-chordal graph. A family $\mathcal{S}(\overline{C_6})$ consisting of one set $\{a,b,c,d,e,f\}$ gives a trivial radius $2 = \lfloor 4/2\rfloor$ acyclic covering of $\overline{C_6}$, and a simple consideration shows that no radius 1 acyclic covering can exist for $\overline{C_6}$ (it is impossible, by simply adding new edges to $\overline{C_6}$, to get a chordal graph in which each maximal clique induces a radius one subgraph of $\overline{C_6}$). In the next subsection we will show that yet an interesting subclass of 4-chordal graphs, namely, the class of chordal bipartite graphs, does admit radius 1 acyclic coverings.

4.1. Collective tree spanners in chordal bipartite graphs. A bipartite graph $G = (X \cup Y, E)$ is *chordal bipartite* if it does not contain any induced cycles of length greater than 4 [48].

For a chordal bipartite graph G, consider a hypergraph $H = (X \cup Y, \{N[y] : y \in Y\})$. In what follows we show that H is an acyclic hypergraph.

LEMMA 4.4. *The 2-section graph $2SEC(H)$ of H is chordal.*

Proof. First notice that any $y \in Y$ is simplicial in $2SEC(H)$ by construction of H and definition of $2SEC(H)$. Assume now, by way of contradiction, that there is an induced cycle C_p of length p, $p \geq 4$, in $2SEC(H)$. Necessarily, all vertices of C_p are from part X of G, since C_p is induced and all vertices from Y are simplicial in $2SEC(H)$. Let $C_p = (x_1, x_2, \ldots, x_p, x_1)$. For any edge $x_i x_{i+1}$ of C_p (including the edge $x_p x_1$), since it is not an edge of G, there must exist a vertex y_i in Y such that both x_i and x_{i+1} are adjacent to y_i in G. Also, since C_p is induced in $2SEC(H)$, y_i is not adjacent to any other vertex of C_p. Therefore, a cycle $(x_1, y_1, x_2, y_2, \ldots, x_p, y_p, x_1)$ of G must be induced. But, since its length is $2p \geq 8$, a contradiction with G being a chordal bipartite graph arises. □

LEMMA 4.5. *The hypergraph $H = (X \cup Y, \{N[y] : y \in Y\})$ is conformal.*

Proof. Let C be a clique of $2SEC(H)$ consisting of p vertices. First, note that, by definitions of H and $2SEC(H)$, the clique C can contain at most one vertex from Y. If C contains a vertex from Y (say $y \in C \cap Y$) then for all $v \in C \setminus \{y\}$, vy is an edge of G, and therefore $C \subseteq N[y]$ must hold. Let now $C \cap Y = \emptyset$. By induction on p we will show that there exists a vertex $y \in Y$ such that $C \subset N[y]$. Since G is connected, any vertex $x \in C \subseteq X$ has a neighbor in Y. Also, by definition of $2SEC(H)$, for any edge uv of $2SEC(H)$ with $u, v \in X$ there must exist a vertex y in Y adjacent to both u and v. Assume now, by induction, that each $p-1$ vertice of C has a common neighbor y in Y. Consider three different vertices a, b and c in C and three corresponding vertices a', b' and c' in Y such that $C \setminus \{a\} \subset N[a']$, $C \setminus \{b\} \subset N[b']$ and $C \setminus \{c\} \subset N[c']$. Since graph G cannot have any induced cycles of length 6, the cycle $(a - b' - c - a' - b - c' - a)$ of G cannot be induced. Without loss of generality, assume that a is adjacent to a' in G. But then, all p vertices of C are contained in $N[a']$. □

Since chordal bipartite graphs form a hereditary class of graphs and, for any chordal bipartite graph $G = (X \cup Y, E)$, a family $\{N[y] : y \in Y\}$ of subsets of $X \cup Y$

satisfies all four conditions of radius 1 acyclic covering, by Theorem 3.3 we have the following theorem.

THEOREM 4.6. *The class of chordal bipartite graphs is* $(1/2, 1)$*-decomposable.*

Hence, by Theorems 2.5 and 2.6, we immediately conclude the following corollaries.

COROLLARY 4.7. *Any chordal bipartite graph* G *with* n *vertices and* m *edges admits an additive* 2*-spanner with at most* $(n-1)\log_2 n$ *edges, and such a sparse spanner can be constructed in* $O(nm \log_2 n)$ *time.*

COROLLARY 4.8. *Any chordal bipartite graph* G *with* n *vertices and* m *edges admits a system* $\mathcal{T}(G)$ *of at most* $\log_2 n$ *collective additive tree* 2*-spanners, and such a system of spanning trees can be constructed in* $O(nm \log_2 n)$ *time.*

Recall that the problem whether a chordal bipartite graph admits a (one) multiplicative tree t-spanner is NP-complete for any $t > 3$ [15]. Also, any chordal bipartite graph G with n vertices admits an additive 4-spanner with at most $2n-2$ edges which is constructable in linear time [21]. Again, it is interesting to know whether a system of constant number of collective additive tree 4-spanners exists for a chordal bipartite graph. We have the following observation on collective additive tree 1-spanners for chordal bipartite graphs.

OBSERVATION 4.9. *There are chordal bipartite graphs on* $2n$ *vertices for which any system of collective additive tree* 1*-spanners will need to have at least* $\Omega(n)$ *spanning trees.*

Proof. Consider the complete bipartite graph $G = K_{n,n}$ on $2n$ vertices (which is clearly a chordal bipartite graph), and let $\mathcal{T}(G)$ be a system of μ collective additive tree 1-spanners of G. Then, for any two adjacent vertices x and y of G there must exist a spanning tree T in $\mathcal{T}(G)$ such that $d_T(x, y) \leq 2$. If $d_T(x, y) = 2$, then a common neighbor z of x and y in G would form a triangle with vertices x and y, which is impossible for $G = K_{n,n}$. Hence, $d_T(x, y) = 1$ must hold. Thus, any edge xy of G is an edge of some tree $T \in \mathcal{T}(G)$. Since there are n^2 graph edges to cover by spanning trees from $\mathcal{T}(G)$, we conclude $\mu \geq n^2/(2n-1) > n/2$. □

4.2. Collective tree spanners in cocomparability graphs. We will use the following definition of cocomparability graphs (see [16, 48, 56]). A graph G is a *cocomparability graph* if it admits a vertex ordering $\sigma = [v_1, v_2, \ldots, v_n]$, called a *cocomparability ordering*, such that, for any $i < j < k$, if v_i is adjacent to v_k, then v_j must be adjacent to v_i or to v_k. According to [56], such an ordering of a cocomparability graph can be constructed in linear time. It is well known also that cocomparability graphs are 4-chordal and they contain all interval graphs, all permutation graphs, and all trapezoid graphs (see, e.g., [16, 48] for the definitions).

Since $\overline{C_6}$ is a cocomparability graph, cocomparability graphs generally do not admit radius 1 acyclic coverings (although, we can show that both the class of permutation graphs and the class of trapezoid graphs do admit radius 1 acyclic coverings [28]). Here we will present a very simple direct proof for the statement that the class of cocomparability graphs is $(1/2, 1)$-decomposable.

THEOREM 4.10. *The class of cocomparability graphs is* $(1/2, 1)$*-decomposable. Moreover, for a given cocomparability graph* G *with* n *vertices and* m *edges a decomposition tree* $\mathcal{BT}(G)$ *can be constructed in* $O(m \log_2 n)$ *time.*

Proof. Let G be a cocomparability graph with a cocomparability ordering $\sigma = [v_1, v_2, \ldots, v_n]$. Consider the closed neighborhood of the vertex $v_{\lceil n/2 \rceil}$. We claim that the graph G' obtained from G by removing vertices of $N[v_{\lceil n/2 \rceil}]$ has no connected components with more that $n/2$ vertices. Indeed, there are no more than $n/2$ vertices

in G which are on the left (analogously, on the right) side of $v_{\lceil n/2 \rceil}$ with respect to σ. Also, if there is an edge connecting vertices v_i and v_j with $i < \lceil n/2 \rceil < j$, then at least one of these vertices must belong to $N[v_{\lceil n/2 \rceil}]$ as σ is a cocomparability ordering. Therefore, each connected component G_s of G' has at most $n/2$ vertices since it consists of vertices which are only on one side of $v_{\lceil n/2 \rceil}$. It is clear also that the ordering σ projected to the vertices of G_s gives a cocomparability ordering of G_s. Hence we can assume by induction that G_s is a $(1/2, 1)$-decomposable graph. □

Hence, we have the following corollaries.

COROLLARY 4.11. *Any cocomparability graph G with n vertices and m edges admits an additive 2-spanner with at most $(n-1)\log_2 n$ edges, and such a sparse spanner can be constructed in $O(m \log_2 n)$ time.*

COROLLARY 4.12. *Any cocomparability graph G with n vertices and m edges admits a system $\mathcal{T}(G)$ of at most $\log_2 n$ collective additive tree 2-spanners, and such a system of spanning trees can be constructed in $O(m \log_2 n)$ time.*

It is known [62] that any cocomparability graph admits a (one) additive tree 3-spanner. In a forthcoming paper [31], using different technique, we show that the result stated in Corollary 4.12 can further be improved. One can show that any cocomparability graph admits a system of two collective additive tree 2-spanners and there are cocomparability graphs which do not have any (one) additive tree 2-spanners. Since the complete bipartite graph $K_{n,n}$ is a cocomparability graph, from the proof of Observation 4.9, we also have the following observation.

OBSERVATION 4.13. *There are cocomparability graphs on n vertices for which any system of collective additive tree 1-spanners will need to have at least $\Omega(n)$ spanning trees.*

5. Collective tree spanners in circular-arc graphs. In this section we describe another way of obtaining a system of few collective additive tree spanners. We demonstrate it on the class of circular-arc graphs.

The *intersection graph* of a family of n sets is the graph where the vertices are the sets, and the edges are the pairs of sets that intersect. Every graph is the intersection graph of some family of sets. A graph $G = (V, E)$ is an *interval graph* if it is the intersection graph of a finite set of intervals (line segments) on a line. A graph G is a *circular-arc graph* if it is the intersection graph of a finite set of arcs on a circle. An interval graph is a special case of a circular-arc graph; it is a circular-arc graph that can be represented with arcs that do not cover the entire circle. Hence, if we remove from a circular-arc graph $G = (V, E)$ a vertex $v \in V$ together with its neighbors, the resulting graph will be interval [48] (see Figure 3 for an illustration).

It is well known that any interval graph admits an additive tree 2-spanner, and such a tree spanner is computable in linear time [61]. On the other hand, for any constant $r \geq 0$, there is a circular-arc graph without any additive tree r-spanner. Indeed, consider an induced cycle C_q on $q \geq 3$ vertices. Clearly, it is a circular-arc graph. Let P be an arbitrary spanning path of C_q and x and y be the end vertices of P. Then, trivially, $d_{C_q}(x, y) = 1$, $d_P(x, y) = q - 1$, i.e., a circular-arc graph C_q does not admit any additive tree $(q-3)$-spanner. In what follows we show that two spanning trees are enough to collectively additively 2-span any circular-arc graph.

Let $G = (V, E)$ be a circular-arc graph, u be its arbitrary vertex, and T_u be a BFS-tree of G rooted at u. Consider an interval graph G^- obtained from G by removing vertices of $N[u]$. For each connected component of G^-, compute its additive tree 2-spanner using a linear time algorithm from [61]. Extend obtained forest to a spanning tree T of the original graph G (see Figure 3 for an illustration).

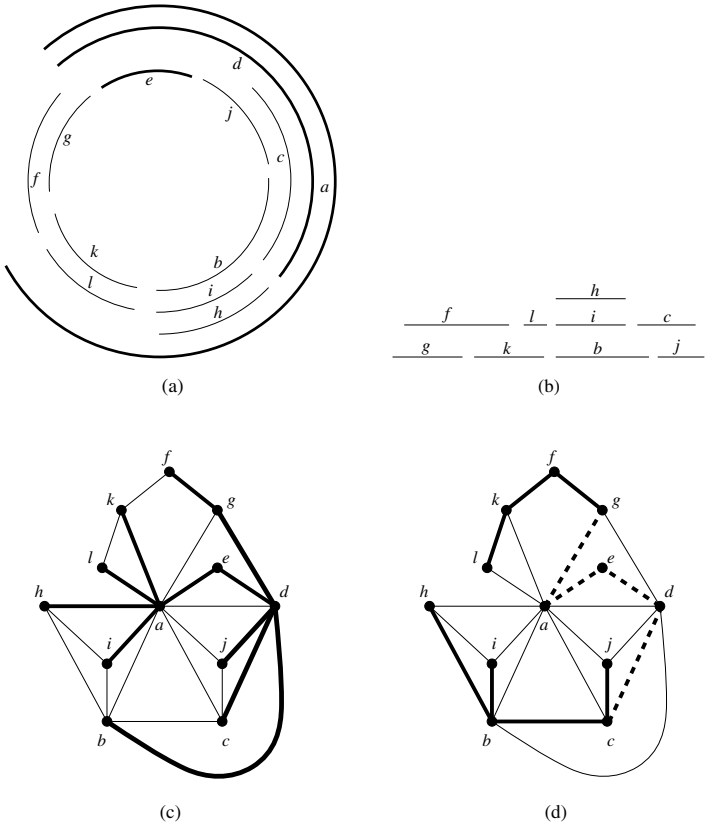


FIG. 3. (a) *A set S of circular-arcs (with arcs from $N[e]$ in bold);* (b) *the set of intervals corresponding to $S \setminus N[e]$;* (c) *the corresponding to S circular-arc graph G with a spanning tree T_e in bold;* (d) *a spanning tree T of G obtained from two local tree 2-spanners.*

LEMMA 5.1. *Spanning trees $\{T, T_u\}$ are collective additive tree 2-spanners of a circular-arc graph G.*

Proof. Let x and y be two arbitrary vertices of G. If there is a shortest path in G connecting vertices x and y and avoiding the neighborhood $N[u]$ of u, then $d_G(x,y) = d_{G^-}(x,y)$ and, by construction of T, $d_T(x,y) \leq d_{G^-}(x,y)+2 = d_G(x,y)+2$ holds. Let now a shortest path P connecting x and y in G intersect $N[u]$, and let v be a vertex from $N[u] \cap P$. Since T_u is a shortest-path tree of G rooted at u, we have $d_G(x,u) = d_{T_u}(x,u)$ and $d_G(u,y) = d_{T_u}(u,y)$. Hence, $d_G(x,y) = d_G(x,v)+d_G(v,y) \geq d_G(x,u) - 1 + d_G(u,y) - 1 = d_{T_u}(x,u) + d_{T_u}(u,y) - 2 \geq d_{T_u}(x,y) - 2$. ∎

Hence, we conclude the following theorem and corollary.

THEOREM 5.2. *Any circular-arc graph G admits a system of two collective additive tree 2-spanners, and such a system of spanning trees can be constructed in linear time.*

COROLLARY 5.3. *Any circular-arc graph G with n vertices and m edges admits an additive 2-spanner with at most $2n-2$ edges, and such a sparse spanner can be constructed in $O(m+n)$ time.*

6. Collective tree spanners and routing labeling schemes. Routing is one of the basic tasks that a distributed network of processors must be able to perform.

A *routing scheme* is a mechanism that can deliver packets of information from any vertex of the network to any other vertex. More specifically, a routing scheme is a distributed algorithm. Each processor in the network has a routing daemon running on it. This daemon receives packets of information and has to decide whether these packets have already reached their destination, and if not, how to forward them towards their destination. Each packet of information has a *header* attached to it. This header contains the address of the destination of the packet, and in some cases, some additional information that can be used to guide the routing of this message towards its destination. Each routing daemon has a local *routing table* at its disposal. It has to decide, based on this table and on the packet header only, whether to pass the packet to its host, or whether to forward the packet to one of its neighbors in the network.

The efficiency of a routing scheme is measured in terms of its *multiplicative stretch*, called *delay*, (or *additive stretch*, called *deviation*), namely, the maximum ratio (or surplus) between the length of a route, produced by the scheme for some pair of vertices, and their distance.

A straightforward approach for achieving the goal of guaranteeing optimal routes is to store a complete routing table in each vertex v in the network, specifying for each destination u the first edge (or an identifier of that edge, indicating the output port) along some shortest path from v to u. However, this approach may be too expensive for large systems since it requires a total of $O(n^2 \log d)$ memory bits in an n-processor network with maximum degree d [41]. Thus, an important problem in large-scale communication networks is the design of routing schemes that produce efficient routes and have relatively low memory requirements (see [3, 24, 35, 49, 57, 60, 68]).

This problem can be approached via localized techniques based on *labeling schemes* [57]. Informally speaking, the routing problem can be presented as requiring us to assign a label to every vertex of a graph. This label can contain the address of the vertex as well as the local routing table. The labels are assigned in such a way that at every source vertex v and given the address of any destination vertex u, one can decide the output port of an outgoing edge of v that leads to u. The decision must be taken locally in v, based solely on the label of v and the address of u.

Following [57], one can give the following formal definition. A family $\Re$ of graphs is said to *have an* $l(n)$ *routing labeling scheme* if there is a *function* L labeling the vertices of each n-vertex graph in $\Re$ with distinct labels of up to $l(n)$ bits, and there exists an efficient algorithm, called the *routing decision*, that given the label of a source vertex v and the label of the destination vertex (the header of the packet), decides in time polynomial in the length of the given labels and using only those two labels, whether this packet has already reached its destination, and if not, to which neighbor of v to forward the packet. Thus, the goal is, for a family of graphs, to find routing labeling schemes with small stretch factor, relatively short labels, and fast routing decision.

To obtain routing schemes for general graphs that use $o(n)$-bit label for each vertex, one has to abandon the requirement that packets are always routed on shortest paths, and settle instead for the requirement that packets are routed on paths with relatively small stretch [3, 4, 24, 35, 60, 68]. A delay-3 scheme that uses labels of size $\tilde{O}(n^{2/3})$ was obtained in [24], and a delay-5 scheme that uses labels of size $\tilde{O}(n^{1/2})$ was obtained in [35].[1] Recently, authors of [68] further improved these results. They presented a routing scheme that uses only $\tilde{O}(n^{1/2})$ bits of memory at each vertex of

[1]Here, $\tilde{O}(f)$ means $O(f \text{ polylog } n)$.

an n-vertex graph and has delay 3. Note that each routing decision takes constant time in their scheme, and the space is optimal, up to logarithmic factors, in the sense that every routing scheme with delay < 3 must use, on some graphs, routing tables of total size $\Omega(n^2)$, and hence $\Omega(n)$ at some vertex (see [39, 42, 45]).

There are many results on optimal (with delay 1) routing schemes for particular graph classes, including complete graphs, grids (alias meshes), hypercubes, complete bipartite graphs, unit interval and interval graphs, trees and 2-trees, rings, tori, unit circular-arc graphs, outerplanar graphs, and squaregraphs. All those graph families admit optimal routing schemes with $O(d \log n)$ labels and $O(\log d)$ routing decision. These results follow from the existence of special so-called *interval routing* schemes for those graphs. We will not discuss details of this scheme here; for precise definitions and an overview of this area, we refer the reader to [41].

Observe that in interval routing schemes the local memory requirement increases with the degree of the vertex. Routing labeling schemes aim at overcoming the problem of large degree vertices. In [40], a shortest-path routing labeling scheme for trees of arbitrary degree and diameter is described that assigns each vertex of an n-vertex tree a $O(\log^2 n/\log\log n)$-bit label. Given the label of a source vertex and the label of a destination it is possible to compute, in constant time, the neighbor of the source that heads in the direction of the destination. A similar result was independently obtained also in [68]. This result for trees was recently used in [32, 33] to design interesting low-deviation routing schemes for chordal graphs and general c-chordal graphs. Reference [32] describes a routing labeling scheme of deviation 2 with labels of size $O(\log^3 n/\log\log n)$ bits per vertex and $O(1)$ routing decision for chordal graphs. Reference [33] describes a routing labeling scheme of deviation $2\lfloor c/2 \rfloor$ with labels of size $O(\log^3 n)$ bits per vertex and $O(\log\log n)$ routing decision for the class of c-chordal graphs.

Our collective additive tree spanners give much simpler and easier to understand means of constructing compact and efficient routing labeling schemes for all (α, r)-decomposable graphs. We simply reduce the original problem to the problem on trees.

Let G be an (α, r)-decomposable graph and let $\mathcal{T}(G) = \{T^1, T^2, \ldots, T^\mu\}$ ($\mu \leq O(\log_2 n)$) be a system of μ collective additive tree $2r$-spanners of G. We can preprocess each tree T^i using the $O(n \log_2 n)$ algorithm from [40] and assign to each vertex v of G a tree label $L^i(v)$ of size $O(\log^2 n/\log\log n)$ bits associated with the tree T^i. Then we can form a label $L(v)$ of v of size $O(\log^3 n/\log\log n)$ bits by concatenating the μ tree labels. We store in $L(v)$ also the string A_v of length $O(\log^2 n)$ bits described in subsection 2.2. Thus, $L(v) := A_v \circ L^1(v) \circ \cdots \circ L^\mu(v)$.

Now assume that a vertex v wants to send a message to a vertex u. Given the labels $L(v)$ and $L(u)$, v first uses their substrings A_v and A_u to find in $\log_2 n$ time an index i such that for tree $T^i \in \mathcal{T}(G)$, $d_{T^i}(v, u) \leq d_G(v, u) + 2r$ holds. Then, v extracts from $L(u)$ the substring $L^i(u)$ and forms a header of the message $H(u) := i \circ L^i(u)$. Now, the initiated message with the header $H(u) = i \circ L^i(u)$ is routed to the destination using the tree T^i: when the message arrives at an intermediate vertex x, vertex x using own substring $L^i(x)$ and the string $L^i(u)$ from the header makes a constant time routing decision.

Thus, the following result is true.

THEOREM 6.1. *Each (α, r)-decomposable graph with n vertices and m edges admits a routing labeling scheme of deviation $2r$ with addresses and routing tables of size $O(\log^3 n/\log\log n)$ bits per vertex. Once computed by the sender in $\log_2 n$ time, headers never change. Moreover, the scheme is computable in $O((p(n) + t(n) + m + n\log_2 n)\log_2 n)$ time, and the routing decision is made in constant time per vertex,*

TABLE 1
Routing labeling schemes obtained for special graph classes via collective additive tree spanners.

Graph class	Scheme construction time	Addresses and routing tables (bits per vertex)	Message initiation time	Routing decision time	Deviation
Chordal	$O(m\log_2 n + n\log_2^2 n)$	$O(\log^3 n/\log\log n)$	$\log_2 n$	$O(1)$	2
Chordal bipartite	$O(nm\log_2 n)$	$O(\log^3 n/\log\log n)$	$\log_2 n$	$O(1)$	2
Cocomparability	$O(m\log_2 n + n\log_2^2 n)$	$O(\log^3 n/\log\log n)$	$\log_2 n$	$O(1)$	2
c-Chordal	$O(n^3\log_2 n)$	$O(\log^3 n/\log\log n)$	$\log_2 n$	$O(1)$	$2\lfloor c/2\rfloor$
Circular-arc	$O(n\log_2 n + m)$	$O(\log^2 n)$	$O(1)$	$O(1)$	2

where $p(n)$ is the time needed to find a balanced and bounded radius separator S and $t(n)$ is the time needed to find a central vertex for S.

Projecting this theorem to the particular graph classes considered in this paper, we obtain the following results summarized in Table 1. For circular-arc graphs, the labels are of size $O(\log^2 n)$ bits per vertex since this size labels are needed to decide in constant time which tree T or T_u is good for routing for given source x and destination y. We will choose tree $T' \in \{T, T_u\}$ such that $d_{T'}(x,y) = \min\{d_T(x,y), d_{T_u}(x,y)\}$. According to [57], in $O(n\log_2 n)$ total time the vertices of an n-vertex tree T can be labeled with labels of up to $O(\log^2 n)$ bits such that, given two labels of two vertices x, y of T, it is possible to compute in constant time the distance $d_T(x,y)$, by merely inspecting the labels of x and y.

7. Extension to the weighted graphs. Although in our previous discussions graph G is assumed (for simplicity) to be unweighted, the obtained results, in slightly modified form, are true even for weighted graphs.

Let $G = (V, E, w)$ be a *weighted graph* with the weight function $w : E \to R^+$. In a weighted graph G, the *length of a path* is the sum of the weights of edges participating in the path. The *distance* $d_G(x,y)$ between vertices x and y is the length of a shortest-length path connecting vertices x and y.

It is easy to see that, if in sections 2–4 we consider shortest path trees instead of BFS-trees, interpret r as an upper bound on the weighted radius of a balanced separator $S \subseteq V$, and denote the maximum edge weight by w, then the following corollaries from the previous results are true.

- Any weighted (α, r)-decomposable graph with n vertices, where r is an upper bound on the weighted radius of a balanced separator, admits a system of at most $\log_{1/\alpha} n$ collective additive tree $2r$-spanners.
- Any weighted c-chordal graph with n vertices admits a system of at most $\log_2 n$ collective additive tree $(2\lfloor c/2\rfloor\mathsf{w})$-spanners.
- Any weighted chordal, chordal bipartite, or cocomparability graph with n vertices admits a system of at most $\log_2 n$ collective additive tree $2\mathsf{w}$-spanners.

8. Conclusion and further developments. In this paper, we introduced a new notion of *collective tree spanners*, and showed that any (α, r)-decomposable graph G with n vertices admits a system of at most $\log_{1/\alpha} n$ collective additive tree $2r$-spanners. As a consequence, we got that any chordal graph, chordal bipartite graph

or cocomparability graph admits a system of at most $\log_2 n$ collective additive tree 2-spanners. We complemented these results by lower bounds, which say that any system of collective additive tree 1-spanners must have $\Omega(\sqrt{n})$ spanning trees for some chordal graphs and $\Omega(n)$ spanning trees for some chordal bipartite graphs and some cocomparability graphs. We also showed that every c-chordal graph admits a system of at most $\log_2 n$ collective additive tree $(2\lfloor c/2 \rfloor)$-spanners and every circular-arc graph admits a system of two collective additive tree 2-spanners. Furthermore, we discussed an application of the collective tree spanners to the problem of designing compact and efficient routing schemes in graphs.

Collective tree spanners can find applications also in designing compact and efficient distance labeling schemes for graphs, defined in [57]. As shown in [57], the vertices of any n-vertex tree T can be labeled with labels of up to $O(\log^2 n)$ bits such that, given two labels of two vertices x, y of T, it is possible to compute in constant time the distance $d_T(x, y)$ by merely inspecting the labels of x and y. Hence, any n-vertex graph G, admitting a system of μ collective additive tree r-spanners, admits a labeling that assigns $O(\mu \log^2 n)$ bit labels to vertices of G such that, given two labels of two vertices x, y of G, it is possible to compute in $O(\mu)$ time an additive r-approximation to the distance $d_G(x, y)$ by merely inspecting the labels of x and y, without using any other information about the graph.

In forthcoming papers [23, 29, 31], we investigate the collective tree spanners problem in other special families of graphs such as homogeneously orderable graphs, AT-free graphs, House–Hole–Domino-free graphs, graphs of bounded tree-width (including series-parallel graphs, outerplanar graphs), graphs of bounded asteroidal number, and others. We show that

- any homogeneously orderable graph admits a system of at most $\log_2 n$ collective additive tree 2-spanners and (one) additive tree 3-spanner,
- any House–Hole–Domino-free graph admits a system of at most $2 \log_2 n$ collective additive tree 2-spanners,
- any AT-free graph admits a system of two collective additive tree 2-spanners,
- any graph whose asteroidal number is bounded by a constant admits a system of a constant number of collective additive tree 3-spanners,
- any graph whose tree-width is bounded by a constant admits a system of at most $O(\log_2 n)$ collective additive tree 0-spanners,
- any graph whose clique-width is bounded by a constant admits a system of at most $O(\log_2 n)$ collective additive tree 2-spanners.

We conclude this paper with a few open questions/problems:

1. What is the complexity of the problem, "Given a graph G and integers μ, r, decide whether G has a system of at most μ collective additive tree r-spanners" for different $\mu \geq 1$, $r \geq 0$ on general graphs and on different restricted families of graphs?
2. What is the best trade-off between the number of trees μ and the additive stretch factor r on planar graphs? (So far, we can state only that any planar graph admits a system of $O(\sqrt{n})$ collective additive tree 0-spanners.)
3. What would be some more applications where collective tree spanners could be useful? The fact that collective tree spanners give a collection of (good) trees might make it easy to adapt many tree algorithms for the graphs that have collective tree r-spanners.

When this paper was already under review for this journal, we learned from A. Gupta that they introduced in [49] a notion of tree covers of graphs which is identical to our notion of collective multiplicative tree spanners. They additionally showed there

that any planar graph admits a system of at most $2\log_2 n$ collective multiplicative tree 3-spanners. This result makes question 2 even more intriguing.

Acknowledgments. We are very grateful to anonymous referees for many useful suggestions.

REFERENCES

[1] I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares, *On sparse spanners of weighted graphs*, Discrete Comput. Geom., 9 (1993), pp. 81–100.
[2] G. Ausiello, A. D'Arti, and M. Moscarini, *Chordality properties on graphs and minimal conceptual connections in sematic data models*, J. Comput. System Sci., 33 (1986), pp. 179–202.
[3] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg, *Improved routing strategies with succinct tables*, J. Algorithms, 11 (1990), pp. 307–341.
[4] B. Awerbuch and D. Peleg, *Routing with polynomial communication-space tradeoff*, SIAM J. Discrete Math., 5 (1992), pp. 151–162.
[5] H.-J. Bandelt and A. Dress, *Reconstructing the shape of a tree from observed dissimilarity data*, Adv. in Appl. Math., 7 (1986), pp. 309-343.
[6] Y. Bartal, *Probabilistic approximations of metric spaces and its algorithmic applications*, in Proceedings of the 37th Annual Symposium on Foundations of Computer Science, IEEE, 1996, pp. 184–193.
[7] Y. Bartal, *On approximating arbitrary metrices by tree metrics*, Proceedings of the 13th Annual ACM Symposium on Theory of Computing, 198, pp. 161-168.
[8] J.-P. Barthélemy and A. Guénoche, *Trees and Proximity Representations*, Wiley, New York, 1991.
[9] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis, *On the desirability of acyclic database schemes*, J. ACM, 30 (1983), pp. 479–513.
[10] C. Berge, *Hypergraphs*, North-Holland, Amsterdam, 1989.
[11] S. Bhatt, F. Chung, F. Leighton, and A. Rosenberg, *Optimal simulations of tree machines*, in Proceedings of the 27th Annual Symposium on Foundations of Computer Science, IEEE, 1986, pp. 274–282.
[12] U. Brandes and D. Handke, *NP–Completeness Results for Minimum Planar Spanners*, preprint, University of Konstanz, Konstanzer Schriften in Mathematik und Informatik, Nr. 16, Germany, 1996.
[13] A. Brandstädt, V. Chepoi, and F. F. Dragan, *Distance approximating trees for chordal and dually chordal graphs*, J. Algorithms, 30 (1999), pp. 166-184.
[14] A. Brandstädt, F. F. Dragan, H.-O. Le, and V. B. Le, *Tree spanners on chordal graphs: Complexity, algorithms, open problems*, in Proceedings of the 13th International Symposium on Algorithms and Computation, Lecture Notes in Comput. Sci. 2518, Springer, Berlin, 2002, pp. 163–174.
[15] A. Brandstädt, F. F. Dragan, H.-O. Le, V. B. Le, and R. Uehara, *Tree spanners for bipartite graphs and probe interval graphs*, in Graph-Theoretic Concepts in Computer Science, Lecture Notes in Comput. Sci. 2880, Springer, Berlin, 2003, pp. 106–118.
[16] A. Brandstädt, V. B. Le, and J. Spinrad, *Graph Classes: A Survey*, SIAM, Philadelphia, 1999.
[17] P. Buneman, *A characterization of rigid circuit graphs*, Discrete Math., 9 (1974), pp. 205–212.
[18] L. Cai, *Tree Spanners: Spanning Trees that Approximate the Distances*, Ph.D. thesis, University of Toronto, 1992.
[19] L. Cai and D. G. Corneil, *Tree spanners*, SIAM J. Discrete Math., 8 (1995), pp. 359–387.
[20] M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin, *Approximating a finite metric by a small number of tree metrics*, in Proceedings of the 39th Annual Symposium on Foundations of Computer Science, IEEE, 1998, pp. 379–388.
[21] V. D. Chepoi, F. F. Dragan, and C. Yan, *Additive spanners for k-chordal graphs*, Proceedings of the 5th Conference on Algorithms and Complexity, Lecture Notes in Comput. Sci. 2653, Springer, Berlin, 2003, pp. 96–107.
[22] L. P. Chew, *There are planar graphs almost as good as the complete graph*, J. Comput. System Sci., 39 (1989), pp. 205–219.
[23] D. G. Corneil, F. F. Dragan, E. Köhler, and C. Yan, *Collective tree 1-spanners for interval graphs*, in Graph-Theoretic Concepts in Computer Science, Lecture Notes in Comput. Sci. 3787, Springer, Berlin, 2005, pp. 151–162.

[24] L. Cowen, *Compact routing with minimum stretch*, in Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, 1999, pp. 255–260.
[25] G. A. Dirac, *On rigid circuit graphs*, Abh. Math. Sem. Univ. Hamburg, 25 (1961), pp. 71–76.
[26] H. N. Djidjev, *On the problem of partitioning planar graphs*, SIAM J. Alg. Discrete Meth., 3 (1982), pp. 229–240.
[27] H. N. Djidjev, *A separator theorem for graphs of fixed genus*, Serdica, 11 (1985), pp. 319–329.
[28] F. F. Dragan and I. Lomonosov, *On compact and efficient routing in certain graph classes*, in Proceedings of the 15th Annual International Symposium on Algorithms and Computation, Lecture Notes in Comput. Sci. 3341, Springer, Berlin, 2004, pp. 402–414.
[29] F. F. Dragan and C. Yan, *Collective Tree Spanners of Homogeneously Orderable Graphs*, in preparation.
[30] F. F. Dragan, C. Yan, and I. Lomonosov, *Collective tree spanners of graphs*, in Proceedings of the 9th Scandinavian Workshop on Algorithm Theory, Lecture Notes in Comput. Sci. 3111, Springer, Berlin, 2004, pp. 64–76.
[31] F. F. Dragan, C. Yan, and D. G. Corneil, *Collective tree spanners and routing in AT-free related graphs*, in Graph-Theoretic Concepts in Computer Science, Lecture Notes in Comput. Sci. 3353, Springer, Berlin, 2004, pp. 68–80.
[32] Y. Dourisboure and C. Gavoille, *Improved compact routing scheme for chordal graphs*, in Proceedings of the 16th International Conference on Distributed Computing, Lecture Notes in Comput. Sci. 2508, Springer, Berlin, 2002, pp. 252–264.
[33] Y. Dourisboure and C. Gavoille, *Tree-Decompositions with Bags of Small Diameter*, Discrete Math., 2003, to appear.
[34] W. Duckwortha and M. Zito, *Sparse hypercube 3-spanners*, Discrete Appl. Math., 103 (2000), pp. 289–295.
[35] T. Eilam, C. Gavoille, and D. Peleg, *Compact routing schemes with low stretch factor*, in Proceedings of the 17th Annual ACM Symposium Prin. Distr. Comput., 1998, pp. 11–20.
[36] R. Fagin, *Degrees of acyclicity for hypergraphs and relational database schemes*, J. ACM, 30 (1983), pp. 514–550.
[37] J. Fakcharoenphol, S. Rao, and K. Talwar, *A tight bound on approximating arbitrary metrics by tree metrics*, in Proceedings of the 35th ACM Symposium on Theory of Computing, 2003, pp. 448–455.
[38] S. P. Fekete and J. Kremer, *Tree spanners in planar graphs*, Discrete Appl. Math., 108 (2001), pp. 85–103.
[39] P. Fraigniaud and C. Gavoille, *Memory requirements for universal routing schemes*, in Proceedings of the 14th Annual ACM Symposium Prin. Distr. Comput., 1995, pp. 223–230.
[40] P. Fraigniaud and C. Gavoille, *Routing in trees*, in Proceedings of the 28th Int. Colloquium on Automata, Languages and Programming, Lecture Notes in Comput. Sci. 2076, Springer, Berlin, 2001, pp. 757–772.
[41] C. Gavoille, *A survey on interval routing schemes*, Theoret. Comput. Sci., 245 (1999), pp. 217–253.
[42] C. Gavoille and M. Gengler, *Space-efficiency of routing schemes of stretch factor three*, J. Parallel and Distr. Comput., 61 (2001), pp. 679–687.
[43] C. Gavoille, M. Katz, N. A. Katz, C. Paul, and D. Peleg, *Approximate distance labeling schemes*, in Proceedings of the 9th Annual European Symposium on Algorithms, Lecture Notes in Comput. Sci. 2161, Springer, Berlin, 2001, pp. 476–487.
[44] C. Gavoille, D. Peleg, S. Pérennes, and R. Raz, *Distance labeling in graphs*, in Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms, 2001, pp. 210–219.
[45] C. Gavoille and S. Pérennès, *Memory requirements for routing in distributed networks*, in Proceedings of the 15th Ann. ACM Symposium on Prin. Distr. Comput., 1996, pp. 125–133.
[46] J. R. Gilbert, J. P. Hutchinson, and R. E. Tarjan, *A separator theorem for graphs of bounded genus*, J. Algorithms, 5 (1984), pp. 391–407.
[47] J. R. Gilbert, D. J. Rose, and A. Edenbrandt, *A separator theorem for chordal graphs*, SIAM J. Alg. Discrete Meth., 5 (1984), pp. 306–313.
[48] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.
[49] A. Gupta, A. Kumar, and R. Rastogi, *Traveling with a Pez dispenser (or, routing issues in MPLS)*, SIAM J. Comput., 34 (2005), pp. 453–474. Appeared also in FOCS IEEE, 2001.
[50] M. Katz, N. A. Katz, and D. Peleg, *Distance labeling schemes for well-separated graph classes*, in Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Comput. Sci. 1770, Springer, Berlin, 2000, pp. 516–528.
[51] H.-O Le and V. B. Le, *Optimal tree 3-spanners in directed path graphs*, Networks, 34 (1999), pp. 81–87.

[52] A. L. Liestman and T. Shermer, *Additive graph spanners*, Networks, 23 (1993), pp. 343–364.
[53] R. J. Lipton and R. E. Tarjan, *A separator theorem for planar graphs*, SIAM J. Appl. Math., 36 (1979), pp. 177–189.
[54] R. J. Lipton and R. E. Tarjan, *Applications of a planar separator theorem*, SIAM J. Comput., 9 (1980), pp. 615–627.
[55] M. S. Madanlal, G. Venkatesan, and C. Pandu Rangan, *Tree* 3*-spanners on interval, permutation and regular bipartite graphs*, Inform. Process. Lett., 59 (1996), pp. 97–102.
[56] R. M. McConnell and J. P. Spinrad, *Linear-time transitive orientation*, in Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms, 1997, pp. 19–25.
[57] D. Peleg, *Distributed Computing: A Locality-Sensitive Approach*, SIAM Monogr. Discrete Math. Appl., SIAM, Philadelphia, 2000.
[58] D. Peleg and A. A. Schäffer, *Graph spanners*, J. Graph Theory, 13 (1989), pp. 99–116.
[59] D. Peleg and J. D. Ullman, *An optimal synchronizer for the hypercube*, in Proceedings of the 6th ACM Symposium on Prin. of Distr. Comput., 1987, pp. 77–85.
[60] D. Peleg and E. Upfal, *A tradeoff between space and efficiency for routing tables*, in Proceedings of the 20th ACM Symposium on the Theory of Computing, 1988, pp. 43–52.
[61] E. Prisner, *Distance approximating spanning trees*, in Proceedings of the 14th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Comput. Sci. 1200, Springer, Berlin, 1997, pp. 499–510.
[62] E. Prisner, D. Kratsch, H.-O. Le, H. Müller, and D. Wagner, *Additive tree spanners*, SIAM J. Discrete Math., 17 (2003), pp. 332–340.
[63] N. Robertson and P. D. Seymour, *Graph minors. Algorithmic aspects of tree-width*, J. Algorithms, 7 (1986), pp. 309–322.
[64] P. H. A. Sneath and R. R. Sokal, *Numerical Taxonomy*, W. H. Freeman, San Francisco, 1973.
[65] J. Soares, *Graph spanners: A survey*, Congr. Numer., 89 (1992), pp. 225–238.
[66] D. L. Swofford and G. J. Olsen, *Phylogeny reconstruction*, in Molecular Systematics, D. M. Hillis and C. Moritz, eds., Sinauer Associates, Sunderland, MA, 1990, pp. 411–501.
[67] R. E. Tarjan and M. Yannakakis, *Simple linear time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*, SIAM J. Comput., 13 (1984), pp. 566–579.
[68] M. Thorup and U. Zwick, *Compact routing schemes*, Proceedings of the 13th Annual ACM Symposium on Par. Alg. and Arch., 2001, pp. 1–10.
[69] G. Venkatesan, U. Rotics, M. S. Madanlal, J. A. Makowsky, and C. Pandu Rangan, *Restrictions of minimum spanner problems*, Inform. and Comput., 136 (1997), pp. 143–164.