Collective Tree Spanners of Graphs

Feodor F. Dragan, Chenyu Yan, and Irina Lomonosov

Department of Computer Science, Kent State University, Kent, Ohio, USA
{dragan, cyan, ilomonos}@cs.kent.edu

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 and any c-chordal graph admits a system of at most $\log_2 n$ collective additive tree $(2\lfloor c/2 \rfloor)$–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.

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. An arbitrary metric space (in particular a finite metric defined by a general graph) might not have enough structure to exploit algorithmically. So, general goal is, for a given graph G, to find a simpler 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 [20,19] 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* of G [17]. 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.

T. Hagerup and J. Katajainen (Eds.): SWAT 2004, LNCS 3111, pp. 64–76, 2004.

Graph spanners have applications in various areas; especially, in distributed systems and communication networks. In [20], 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. Unfortunately, the problem of determining, for a given graph G and two integers $t, m \geq 1$, whether G has a multiplicative t-spanner with m or fewer edges, is NP-complete (see [19]).

The sparsest spanners are tree spanners. As it was shown in [18], 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 [6]. 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: chordal graphs [3] and chordal bipartite graphs [4].

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) [8,21]. However, as it was shown in [19], 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 [8], 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.

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. 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 work [1] of Bartal and subsequent work [7]. For example, motivated by Bartal's work on probabilistic approximation of general metrics with tree metrics, [7] 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 (see [1,7] 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 [19] and [8] on 3-spanners and answers the question posed in [8] whether chordal graphs admit additive 2-spanners with $O(n \log n)$ edges.

In section 5 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 5). 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 [13] (see also [12] for the case of chordal graphs). We conclude the paper with Section 6 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., $rad_G(S) = min_{v \in V}\{max_{u \in S}\{d_G(u,v)\}\}$ and $diam_G(S) = max_{u,v \in S}\{d_G(u,v)\}$. 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. For example, bounded size balanced separators and bounded diameter balanced separators were recently employed in [16] 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 non-negative integer. We say that an n-vertex graph $G = (V, E)$ is (α, r)*–decomposable* if the following three conditions hold for G:

Balanced Separator condition - there exists a set $S \subseteq V$ of vertices in G whose removal leaves no connected component with more than αn vertices;

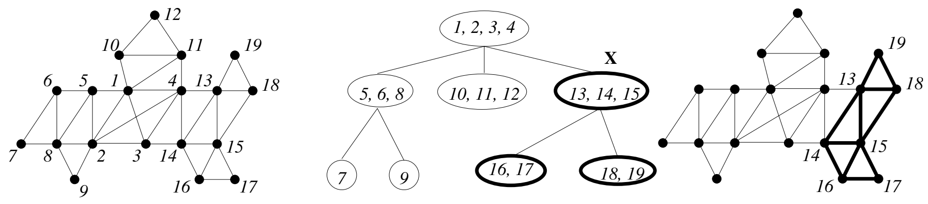
Bounded Separator-Radius condition - $rad_G(S) \leq r$, i.e., there exists a vertex c in G (called *a central vertex for* S) such that $d_G(v,c) \leq r$ for any $v \in S$;

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.

Using the first and third conditions, 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 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



(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.

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 $(X_0, X_1, \dots, X_t)$ 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, $X_0, X_1, \dots, X_t$ are the common ancestors of $S(x)$ and $S(y)$). The following lemmata are crucial to all our subsequent results.

Lemma 1. *Any path $P^G_{x,y}$, connecting vertices x and y in G, contains a vertex from $X_0 \cup X_1 \cup \dots \cup X_t$.*

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 $(X_0, X_1, \dots, X_t)$ with the smallest index such that $SP^G_{x,y} \bigcap X_i \neq \emptyset$ in G. Then, the following lemma holds.

Lemma 2. *We have $d_G(x,y) = d_{G'}(x,y)$, where $G' := G(\downarrow X_i)$.*

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 $d_{G'}(v,c) \le r$ 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, y.

Lemma 3. *We have $d_{T'}(x,y) \le d_G(x,y) + 2r$.*

Let now $B^i_1, \dots, B^i_{p_i}$ be the nodes on depth i of the tree $\mathcal{BT}(G)$. For each subgraph $G^i_j := G(\downarrow B^i_j)$ of G ($i = 0, 1, \dots, depth(\mathcal{BT}(G))$, $j = 1, 2, \dots, p_i$), 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. 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 3 the following general result can be deduced.

Theorem 1. *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 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. *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 1, 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 $d_{T^{i'}}(x, y) \le d_G(x, y) + 2r$. Thus, we have

Theorem 3. *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.

3 Acyclic Hypergraphs, Chordal Graphs, and (α, r)–Decomposable Graphs

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 non-empty 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 [2].

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. A vertex v of a graph G is called *simplicial* if its neighborhood $N(v)$ form a clique in G.

Theorem 4. *(see [2,5]) 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 $G = (V, E)$.

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 5. *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.*

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 5 and Theorems 2 and 3, we immediately conclude

Corollary 1. *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 2. *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 1 improves a known result of Peleg and Schäffer on sparse spanners of chordal graphs. In [19], 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 [8], 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 2, 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 $t > 3$ [3].

Peleg and Schäffer showed also in [19] 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 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.*

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

A balanced separator of radius at most $\lfloor c/2 \rfloor$ of a c-chordal graph G on n vertices can be found in $O(n^3)$ time. Thus, from Theorems 2 and 3, we conclude

Corollary 3. *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. *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 $\overline{C_6}$).

Next 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. A bipartite graph $G = (X \cup Y, E)$ is *chordal bipartite* if it does not contain any induced cycles of length greater than 4.

For a chordal bipartite graph G, consider a hypergraph $H = (X \cup Y, \{N[y] : y \in Y\})$. In full version we show that H is an acyclic hypergraph. 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 5 we have

Theorem 8. *The class of chordal bipartite graphs is* $(1/2, 1)$*-decomposable.*

Another interesting subclass of 4-chordal graphs is the class of cocomparability graphs. It is well-known that cocomparability graphs contain all interval graphs, all permutation graphs and all trapezoid graphs (see, e.g., [5] 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 [9]). In full version we present a very simple direct proof for the statement that the class of cocomparability graphs is $(1/2, 1)$-decomposable.

Theorem 9. *The class of cocomparability graphs is* $(1/2, 1)$*-decomposable.*

Corollary 5. *Any chordal bipartite graph or 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 $O(nm \log_2 n)$ time for chordal bipartite graphs and in $O(m \log_2 n)$ time for cocomparability graphs.*

Corollary 6. *Any chordal bipartite graph or 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 $O(nm \log_2 n)$ time for chordal bipartite graphs and in $O(m \log_2 n)$ time for cocomparability graphs.*

Recall that the problem whether a chordal bipartite graph admits a (one) multiplicative tree t-spanner is NP-complete for any $t > 3$ [4]. 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 [8]. Again, it is interesting to know whether a system of constant number of collective additive tree 4–spanners exists for a chordal bipartite graph.

It is known [21] that any cocomparability graph admits an (one) additive tree 3-spanner. In a forthcoming paper [11], using different technique, we show that the result stated in Corollary 6 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-spanner.

We have the following observation on collective additive tree 1–spanners for chordal bipartite graphs and cocomparability graphs.

Observation 10. *There are chordal bipartite graphs and 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 and Routing Labeling Schemes

An important problem in large scale communication networks is the design of routing schemes that produce efficient routes and have relatively low memory requirements. Following [18], 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. 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. Thus, the goal is, for a family of graphs, to find a routing labeling scheme 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. Recently, authors of [22] presented a routing scheme that uses $\tilde{O}(n^{1/2})$ bits of memory at each vertex of 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 a logarithmic factor, in the sense that every routing scheme with delay < 3 must use, on some graphs, routing labels of total size $\Omega(n^2)$, and hence $\Omega(n)$ at some vertex (see [15]).

In [14,22], 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. This result for trees was recently used in [12,13] to design interesting low deviation routing schemes for chordal graphs and general c-chordal graphs. [12] 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. [13] 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. The following result is true.

Theorem 11. *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. Moreover, once computed by the sender in $log_2 n$ time, headers never change, and the routing decision is made in constant time per vertex.*

Projecting this theorem to the particular graph classes considered in this paper, we obtain the following result:

- Any c-chordal graph (resp., chordal, chordal bipartite or cocomparability graph) admits a routing labeling scheme of deviation $2\lfloor c/2 \rfloor$ (resp., of deviation 2) with addresses and routing tables of size $O(\log^3 n/\log\log n)$ bits per vertex. Moreover, once computed by the sender in $log_2 n$ time, headers never change, and the routing decision is made in constant time per vertex.

6 Further Developments

In forthcoming papers [9,10,11], we extend the method described in Section 2 and apply it to other families of graphs such as homogeneously orderable graphs, AT-free graphs, graphs of bounded tree-width (including series-parallel graphs, outerplanar graphs), graphs of bounded asteroidal number, and others. We show

- any homogeneously orderable graph admits a system of at most $\log_2 n$ collective additive tree 2–spanners,
- any AT-free graph admits a system of two collective additive tree 2–spanners,
- any graph with bounded by a constant asteroidal number admits a system of a constant number of collective additive tree 3–spanners,
- any graph of bounded by a constant tree-width admits a system of at most $O(\log_2 n)$ collective additive tree 0–spanners.

Note that, although the class of homogeneously orderable graphs is not hereditary, our ideas still applicable.

We conclude this paper with two open questions:

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 $O(\sqrt{n}\log_2 n)$ collective additive tree 0-spanners.)

References

1. Y. Bartal, On approximating arbitrary metrices by tree metrics, Proceedings of the *13th Annual ACM Symposium on Theory of Computing,* pp. 161-168, 1998.
2. C. Berge, Hypergraphs, *North Holland*, 1989.
3. A. Brandstädt, F.F. Dragan, H.-O. Le, and V.B. Le, Tree Spanners on Chordal Graphs: Complexity, Algorithms, Open Problems, *Proceedings of the 13th International Symposium on Algorithms and Computation (ISAAC'02)*, November 2002, Springer, Lecture Notes in Computer Science 2518, pp. 163–174.
4. A. Brandstädt, F.F. Dragan, H.-O. Le, V.B. Le and R. Uehara, Tree spanners for bipartite graphs and probe interval graphs, In *Proceedings of the 29th International Workshop "Graph-Theoretic Concepts in Computer Science" (WG '03),* June 2003, Elspeet, the Netherlands, Springer, Lecture Notes in Computer Science 2880, pp. 106-118.
5. A. Brandstädt, V.B. Le and J. Spinrad, Graph Classes: A Survey, *SIAM,* Philadelphia, 1999.
6. L. Cai and D.G. Corneil, Tree spanners, *SIAM J. Disc. Math.,* 8 (1995), 359–387.
7. M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin, Approximating a Finite Metric by a Small Number of Tree Metrics, Proceedings of the *39th Annual Symposium on Foundations of Computer Science*, pp. 379–388, 1998.
8. V. D. Chepoi, F. F. Dragan, and C. Yan, Additive Spanners for k-Chordal Graphs, In *Proceedings of the 5th Conference on Algorithms and Complexity (CIAC 2003)*, May 28-30, 2003, Rome, Italy, Springer, Lecture Notes in Computer Science 2653, pp. 96-107.
9. F.F. Dragan and I. Lomonosov, Compact and efficient routing schemes for special graph classes, *in preparation,* 2004.
10. F.F. Dragan and C. Yan, Collective tree spanners of homogeneously orderable graphs, *in preparation,* 2004.
11. F.F. Dragan, C. Yan and D.G. Corneil, Collective tree spanners and routing in AT-free related graphs, *in preparation,* 2004.
12. Y. Dourisboure and C. Gavoille, Improved Compact Routing Scheme for Chordal Graphs, In proceedings of the *16th International Conference on Distributed Computing (DISC 2002)*, Toulouse, France, October 28-30, 2002, *Lecture Notes in Computer Science* 2508, Springer, pp. 252-264.
13. Y. Dourisboure and C. Gavoille, Tree-length of graphs, *manuscript,* 2003.
14. P. Fraigniaud and C. Gavoille, Routing in Trees, in *Proc. 28th Int. Colloquium on Automata, Languages and Programming (ICALP 2001),* Lecture Notes in Computer Science 2076, 2001, pp. 757–772.
15. C. Gavoille and M. Gengler, Space-efficiency of routing schemes of stretch factor three, *J. Parallel and Distr. Comput.,* 61 (2001), 679-687.

16. 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 (STACS 2000)*, Lille, France, February 2000, *Lecture Notes in Computer Science* 1770, Springer, 2000, pp. 516–528.
17. A.L. Liestman and T. Shermer, Additive graph spanners, *Networks,* 23 (1993), 343–364.
18. D. Peleg, Distributed Computing: A Locality-Sensitive Approach, *SIAM Monographs on Discrete Math. Appl.*, SIAM, Philadelphia, 2000.
19. D. Peleg, and A.A. Schäffer, Graph Spanners, *J. Graph Theory*, 13 (1989), 99-116.
20. D. Peleg and J.D. Ullman, An optimal synchronizer for the hypercube, *in Proc. 6th ACM Symposium on Principles of Distributed Computing,* Vancouver, 1987, 77–85.
21. E. Prisner, D. Kratsch, H.-O. Le, H. Müller, and D. Wagner, Additive tree spanners, *SIAM Journal on Discrete Mathematics*, 17 (2003), 332–340.
22. M. Thorup and U. Zwick, Compact routing schemes, in *Proc. 13th Ann. ACM Symp. on Par. Alg. and Arch. (SPAA 2001)*, ACM 2001, pp. 1–10.