Additive Spanners for k-Chordal Graphs

Victor D. Chepoi[1], Feodor F. Dragan[2], and Chenyu Yan[2]

[1] Laboratoire d'Informatique Fondamentale, Université Aix-Marseille II, France.
chepoi@lim.univ-mrs.fr
[2] Department of Computer Science, Kent State University, Kent, OH 44242, USA
dragan@cs.kent.edu, cyan1@kent.edu

Abstract. In this paper we show that every chordal graph with n vertices and m edges admits an additive 4-spanner with at most $2n-2$ edges and an additive 3-spanner with at most $O(n \cdot \log n)$ edges. This significantly improves results of Peleg and Schäffer from [Graph Spanners, *J. Graph Theory*, 13(1989), 99-116]. Our spanners are additive and easier to construct. An additive 4-spanner can be constructed in linear time while an additive 3-spanner is constructable in $O(m \cdot \log n)$ time. Furthermore, our method can be extended to graphs with largest induced cycles of length k. Any such graph admits an additive $(k+1)$-spanner with at most $2n-2$ edges which is constructable in $O(n \cdot k + m)$ time.

Classification: Algorithms, Sparse Graph Spanners

1 Introduction

Let $G = (V, E)$ be a connected graph with n vertices and m edges. The *length* of a path from a vertex v to a vertex u in G is the number of edges in the path. The distance $d_G(u, v)$ between vertices u and v is the length of a shortest (u, v)-path of G. We say that a graph $H = (V, E')$ is *an additive r-spanner* (*a multiplicative t-spanner*) of G, if $E' \subseteq E$ and $d_H(x, y) - d_G(x, y) \leq r$ $(d_H(x, y)/d_G(x, y) \leq t$, respectively) holds for any pair of vertices $x, y \in V$ (here $t \geq 1$ and $r \geq 0$ are real numbers). We refer to r (to t) as the *additive* (respectively, *multiplicative*) *stretch factor* of H. Clearly, every additive r-spanner of G is a multiplicative $(r+1)$-spanner of G (but not vice versa).

There are many applications of spanners 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. 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 [21]. Unfortunately, the problem of determining, for a given graph G and two integers $t, m \geq 1$, whether G has a t-spanner with m or fewer edges, is NP-complete (see [19]).

The sparsest spanners are tree spanners. Tree spanners occur in biology [2], and as it was shown in [18], they can be used as models for broadcast operations.

R. Petreschi et al. (Eds.): CIAC 2003, LNCS 2653, pp. 96–107, 2003.

Multiplicative tree t-spanners were considered in [7]. It was shown that, for a given graph G, the problem to decide whether G has a multiplicative tree t–spanner is NP–complete for any fixed $t \geq 4$ and is linearly solvable for $t = 1, 2$ (the status of the case $t = 3$ is open for general graphs). Also, in [10], NP–completeness results were presented for tree spanners on planar graphs.

Many 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 admit additive tree r-spanners and/or multiplicative tree t-spanners for sufficiently small r and t (see [3,6,13,14,16,22,23]). We refer also to [1,5,4,6,7,15,17,18,19,24] for more background information on tree and general sparse spanners.

In this paper we are interested in finding sparse spanners with small additive stretch factors in chordal graphs and their generalizations. A graph G is *chordal* [12] if its largest induced (chordless) cycles are of length 3. A graph is *k-chordal* if its largest induced cycles are of length at most k.

The class of chordal graphs does not admit good tree spanners. As it was mentioned in [22,23], H.-O. Le and T.A. McKee have independently showed that for every fixed integer t there is a chordal graph without tree t–spanners (additive as well as multiplicative). Recently, Brandstädt et al. [4] have showed that, for any $t \geq 4$, the problem to decide whether a given chordal graph G admits a multiplicative tree t-spanner is NP-complete even when G has the diameter at most $t + 1$ (t is even), respectively, at most $t + 2$ (t is odd). Thus, the only hope for chordal graphs is to get sparse (with $O(n)$ edges) small stretch factor spanners. Peleg and Schäffer have already showed in [19] that any chordal graph admits a multiplicative 5-spanner with at most $2n - 2$ edges and a multiplicative 3-spanner with at most $O(n \cdot \log n)$ edges. Both spanners can be constructed in polynomial time.

In this paper we improve those results. We show that every chordal graph admits an additive 4-spanner with at most $2n-2$ edges and an additive 3-spanner with at most $O(n \cdot \log n)$ edges. Our spanners are not only additive but also easier to construct. An additive 4-spanner can be constructed in linear time while an additive 3-spanner is constructable in $O(m \cdot \log n)$ time. Furthermore, our method can be extended to all k-chordal graphs. Any such graph admits an additive $(k+1)$-spanner with at most $2n - 2$ edges which is constructable in $O(n \cdot k + m)$ time. Note that the method from [19] essentially uses the characteristic *clique trees* of chordal graphs and therefore cannot be extended (at least directly) to general k-chordal graphs for $k \geq 4$. In obtaining our results we essentially relayed on ideas developed in papers [3], [8], [9] and [19].

2 Preliminaries

All graphs occurring in this paper are connected, finite, undirected, loopless, and without multiple edges. For each integer $l \geq 0$, let $B_l(u)$ denote the *ball* of radius l centered at u: $B_l(u) = \{v \in V : d_G(u,v) \leq l\}$. Let $N_l(u)$ denote the *sphere* of radius l centered at u: $N_l(u) = \{v \in V : d_G(u,v) = l\}$. $N_l(u)$ is called also the *lth neighborhood* of u. A *layering* of G with respect to some vertex u is a partition of V into the spheres $N_l(u)$, $l = 0, 1, \ldots$. By $N(u)$ we denote the *neighborhood* of u, i.e., $N(u) = N_1(u)$. More generally, for a subset $S \subseteq V$ let $N(S) = \bigcup_{u \in S} N(u)$.

Let $\sigma = [v_1, v_2, \ldots, v_n]$ be any ordering of the vertex set of a graph G. We will write $a < b$ whenever in a given ordering σ vertex a has a smaller number than vertex b. Moreover, $\{a_1, \cdots, a_l\} < \{b_1, \cdots, b_k\}$ is an abbreviation for $a_i < b_j$ ($i = 1, \cdots, l$; $j = 1, \cdots, k$). In this paper, we will use two kind of orderings, namely, BFS-orderings and LexBFS-orderings.

In a *breadth-first search (BFS)*, started at vertex u, the vertices of a graph G with n vertices are numbered from n to 1 in decreasing order. The vertex u is numbered by n and put on an initially empty queue of vertices. Then a vertex v at the head of the queue is repeatedly removed, and neighbors of v that are still unnumbered are consequently numbered and placed onto the queue. Clearly, BFS operates by proceeding vertices in layers: the vertices closest to the start vertex are numbered first, and most distant vertices are numbered last. BFS may be seen to generate a rooted tree T with vertex u as the root. We call T the *BFS-tree* of G. A vertex v is the *father* in T of exactly those neighbors in G which are inserted into the queue when v is removed. An ordering σ generated by a BFS will be called a *BFS-ordering* of G. Denote by $f(v)$ the father of a vertex v with respect to σ. The following properties of a BFS-ordering will be used in what follows.

(P1) If $x \in N_i(u), y \in N_j(u)$ and $i < j$, then $x > y$ in σ.

(P2) If $v \in N_q(u) (q > 0)$ then $f(v) \in N_{q-1}(u)$ and $f(v)$ is the vertex from $N(v)$ with the largest number in σ.

(P3) If $x > y$, then either $f(x) > f(y)$ or $f(x) = f(y)$.

Lexicographic breadth-first search (LexBFS), started at a vertex u, orders the vertices of a graph by assigning numbers from n to 1 in the following way. The vertex u gets the number n. Then each next available number k is assigned to a vertex v (as yet unnumbered) which has lexically largest vector $(s_n, s_{n-1}, \ldots, s_{k+1})$, where $s_i = 1$ if v is adjacent to the vertex numbered i, and $s_i = 0$ otherwise. An ordering of the vertex set of a graph generated by LexBFS we will call a *LexBFS-ordering*. Clearly any LexBFS-ordering is a BFS-ordering (but not conversely). Note also that for a given graph G, both a BFS-ordering and a LexBFS-ordering can be generated in linear time [12]. LexBFS-ordering has all the properties of the BFS-ordering. In particular, we can associate a tree T rooted at v_n with every LexBFS-ordering $\sigma = [v_1, v_2, \ldots, v_n]$ simply connecting every vertex v ($v \neq v_n$) to its neighbor $f(v)$ with the largest number in σ. We call this tree a *LexBFS-tree* of G rooted at v_n and vertex $f(v)$ *the father* of v in T.

3 Spanners for Chordal Graphs

3.1 Additive 4-Spanners with $O(n)$ Edges

For a chordal graph $G = (V, E)$ and a vertex $u \in V$, consider a BFS of G started at u and let $q = max\{d_G(u, v) : v \in V\}$. For a given k, $0 \leq k \leq q$, let S_1^k, S_2^k,..., $S_{p_k}^k$ be the connected components of a subgraph of G induced by the kth neighborhood of u. In [3], there was defined a graph Γ whose vertices are the connected components S_i^k, $k = 0, 1, \ldots, q$ and $i = 1, \ldots, p_k$. Two vertices S_i^k, S_j^{k-1} are adjacent if and only if there is an edge of G with one end in S_i^k and another end in S_j^{k-1}. Before we describe our construction of the additive 4-spanner $H = (V, E')$ for a chordal graph G, first we recall two important lemmas.

Lemma 1. *[3] Let G be a chordal graph. For any connected component S of the subgraph of G induced by $N_k(u)$, the set $N(S) \cap N_{k-1}(u)$ induces a complete subgraph.*

Lemma 2. *[3] Γ is a tree.*

Now, to construct H, we choose an arbitrary vertex $u \in V$ and perform a Breadth-First-Search in G started at u. Let $\sigma = [v_1, \ldots, v_n]$ be a BFS-ordering of G. The construction of H is done according to the following algorithm (for an illustration see Figure 1).

PROCEDURE 1. Additive 4-spanners for chordal graphs

Input: A chordal graph $G = (V, E)$ with BFS-ordering σ, and connected components S_1^k, S_2^k, ..., $S_{p_k}^k$ for any k, $0 \leq k \leq q$, where $q = max\{d_G(u, v) : v \in V\}$.

Output: A spanner $H = (V, E')$ of G.

Method:

$E' = \emptyset$;
for $k = q$ **downto** 1 **do**
  for $j = 1$ **to** p_k **do**
    $M = \emptyset$;
    for each vertex $v \in S_j^k$ add edge $vf(v)$ to E' and vertex $f(v)$ to M;
    pick vertex $c \in M$ with the minimum number in σ;
    for every vertex $x \in M \setminus \{c\}$ add edge xc to E';
return $H = (V, E')$.

Lemma 3. *If G has n vertices, then H contains at most $2n - 2$ edges.*

Proof: The edge set of H consists of two sets E_1 and E_2, where E_1 are those edges connecting two vertices between two different layers (edges of type $vf(v)$) and E_2 are those edges which have been used to build a star for a clique M inside a layer (edges of type $cf(v)$). Obviously, E_1 has exactly $n - 1$ edges; actually,

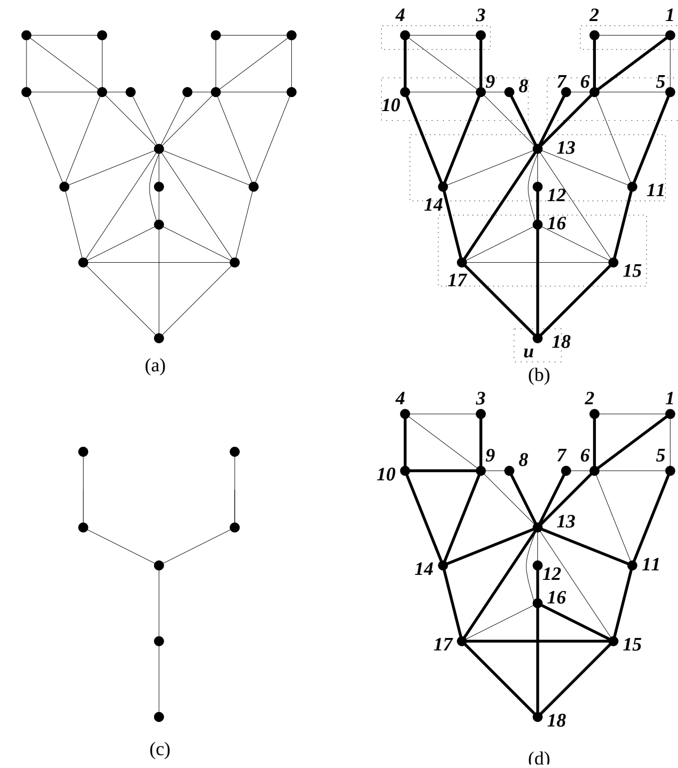

Fig. 1. (a) A chordal graph G. (b) A BFS-ordering σ, BFS-tree T associated with σ and a layering of G. (c) The tree Γ of G associated with that layering. (d) Additive 4-spanner (actually, additive 3-spanner) H of G constructed by PROCEDURE 1 (5 edges are added to the BFS-tree T).

they are the edges of the BFS-tree of G. For each connected component S_i^k of size s, we have at most s vertices in M. Therefore, while proceeding component S_i^k, at most $s-1$ edges are added to E_2. The total size of all the connected components is at most n, so E_2 contains at most $n-1$ edges. Hence, the graph H contains at most $2n-2$ edges. □

Lemma 4. *H is an additive 4-spanner for G.*

Proof: Consider nodes S_i^k and S_j^l of the tree Γ (rooted at $S_1^0 = \{u\}$) and their lowest common ancestor S_m^p in Γ. For any two vertices $x \in S_i^k$ and $y \in S_j^l$ of G, we have $d_G(x,y) \geq k-p+l-p$, since any path of G connecting x and y must pass S_m^p.

From our construction of H (for every vertex v of G the edge $vf(v)$ is present in H), we can easily show that there exist vertices $x', y' \in S_m^p$ such that $d_H(x,x') = k-p$, $d_H(y,y') = l-p$. Hence we only need to show that $d_H(x',y') \leq 4$. If $x' = y'$ then we are done. If vertices x' and y' are distinct, then by Lemma 1, $N(S^p_m) \cap N_{p-1}(u)$ is a clique of G. According to the Procedure 1, fathers of both vertices x' and y' are in M and they are connected in H by a path of length at most 2 via vertex c of M. Therefore, $d_H(x', y') \leq d_H(x', f(x')) + d_H(f(x'), f(y')) + d_H(f(y'), y') \leq 1 + 2 + 1 = 4$. This concludes our proof. □

We can easily show that the bounds given in Lemma 4 are tight. For a chordal graph presented in Figure 2, we have $d_G(y, b) = 1$. The spanner H of G constructed by our method is shown in bold edges. In H we have $d_H(y, b) = 5$. Therefore, $d_H(y, b) - d_G(y, b) = 4$.

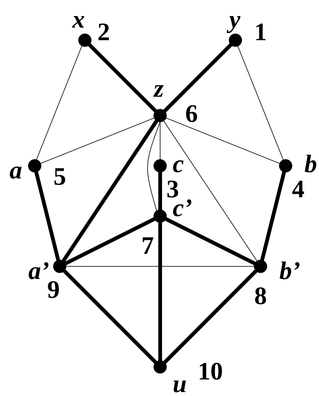

Fig. 2. A chordal graph with a BFS-ordering which shows that the bounds given in Lemma 4 are tight. We have $d_H(y, b) - d_G(y, b) = 4$ and $d_H(y, b)/d_G(y, b) = 5$.

Lemma 5. *H can be constructed in linear $O(n + m)$ time.*

Combining Lemmas 4-5 we get the following result.

Theorem 1. *Every n-vertex chordal graph $G = (V, E)$ admits an additive 4-spanner with at most $2n - 2$ edges. Moreover, such a sparse spanner of G can be constructed in linear time.*

Notice that any additive 4-spanner is a multiplicative 5-spanner. As we mentioned earlier the existence of multiplicative 5-spanners with at most $2n-2$ edges in chordal graphs was already shown in [19], but their method of constructing such spanners is more complicated than ours and can take more than linear time.

3.2 Additive 3-Spanners with $O(n \cdot \log n)$ Edges

To construct an additive 3-spanner for a chordal graph $G = (V, E)$, first we get a LexBFS-ordering σ of the vertices of G (see Figure 3). Then, we construct an additive 4-spanner $H = (V, E_1 \bigcup E_2)$ for G using the algorithm from Section 3.1. Finally, we update H by adding some more edges. In what follows, we will need the following known result.

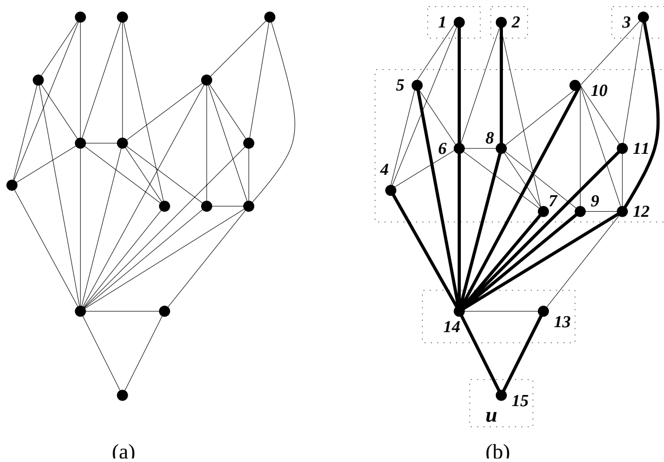

Fig. 3. (a) A chordal graph G. (b) A LexBFS-ordering σ, LexBFS-tree associated with σ and a layering of G.

Theorem 2. *[11] Every n-vertex chordal graph G contains a maximal clique C such that if the vertices in C are deleted from G, every connected component in the graph induced by any remaining vertices is of size at most $n/2$.*

An $O(n+m)$ algorithm for finding such a separating clique C is also given in [11].

As before, for a given k, $0 \leq k \leq q$, let S_1^k, $S_2^k, \ldots, S_{p_k}^k$ be the connected components of a subgraph of G induced by the kth neighborhood of u. For each connected component S_i^k (which is obviously a chordal graph), we run the following algorithm which is similar to the algorithm in [19] (see also [18]), where a method for construction of a multiplicative 3-spanner for a chordal graph is described. The only difference is that we run that algorithm on every connected component from each layer of G instead of on the whole graph G. For the purpose of completeness, we present the algorithm here (for an example see Figure 4).

PROCEDURE 2. A balanced clique tree for a connected component S_i^k

Input: A subgraph Q of G induced by a connected component S_i^k.
Output: A balanced clique tree for Q.

Method:

find a maximum separating clique C of the graph Q
  as prescribed in Theorem 2;
suppose C partitions the rest of Q into connected
  components $\{Q_1, \ldots, Q_r\}$;
for each Q_i, construct a balanced clique tree
  $T(Q_i)$ recursively;
construct $T(Q)$ by taking C to be the root and
  connecting the root of each tree $T(Q_i)$ as a child of C.

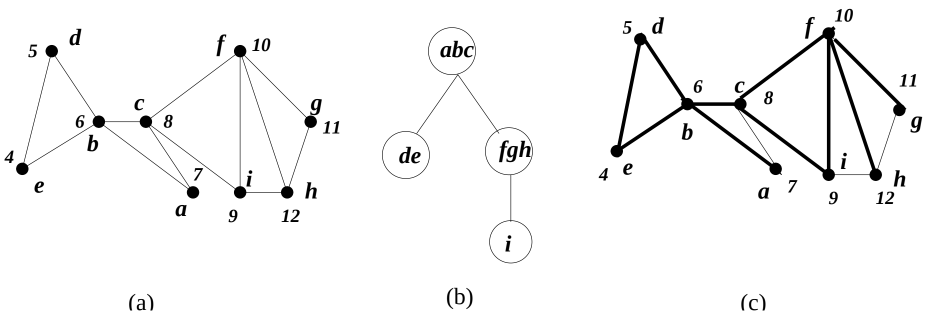

Fig. 4. (a) A chordal graph induced by set S_1^2 of the graph G presented in Figure 3, (b) its balanced clique tree and (c) edges of $E_3(S_1^2) \bigcup E_4(S_1^2)$.

The nodes of the final balanced tree for S_i^k (denote it by $T(S_i^k)$) represent a certain collection of disjoint cliques $\{C_i^k(1), \dots C_i^k(s_i^k)\}$ that cover entire set S_i^k (see Figure 4 for an illustration). For each clique $C_i^k(j)$ $(1 \le j \le s_i^k)$ we build a star centered at its vertex with the minimum number in LexBFS-ordering σ. We use $E_3(i,k)$ to denote this set of star edges. Evidently, $|E_3(i,k)| \le |S_i^k| - 1$.

Consider a clique $C_i^k(j)$ in S_i^k. For each vertex v of $C_i^k(j)$ and each clique $C_i^k(j')$ on the path of balanced clique tree $T(S_i^k)$ connecting node $C_i^k(j)$ with the root, if v has a neighbor in $C_i^k(j')$, then select one such neighbor w and put the edge vw in set $E_4(i,k)$ (initially $E_4(i,k)$ is empty). We do this for every clique $C_i^k(j)$, $j \in \{1, \dots, s_i^k\}$. Since the depth of the tree $T(S_i^k)$ is at most $log_2|S_i^k| + 1$ (see [19], [18]), any vertex v from S_i^k may contribute at most $log_2|S_i^k|$ edges to $E_4(i,k)$. Therefore, $|E_4(i,k)| \le |S_i^k| \cdot log_2|S_i^k|$.

Define now two sets of edges in G, namely,

$$E_3 = \bigcup_{k=1}^{q} \bigcup_{i=1}^{p_k} E_3(i,k), \qquad E_4 = \bigcup_{k=1}^{q} \bigcup_{i=1}^{p_k} E_4(i,k),$$

and consider a spanning subgraph $H^* = (V, E_1 \bigcup E_2 \bigcup E_3 \bigcup E_4)$ of G (see Figure 5). Recall that $E_1 \bigcup E_2$ is the set of edges of an additive 4-spanner H constructed for G by PROCEDURE 1 (see Section 3.1).

From what has been established above one can easily deduce that $|E_3| \le n-1$ and $E_4| \le n \cdot \log_2 n$, thus yielding the following result.

Lemma 6. *If G has n vertices, then H^* has at most $O(n \cdot \log n)$ edges.*

To prove that H^* is an additive 3-spanner for G, we will need the following auxiliary lemmas.

Lemma 7. *[12] Let G be a chordal graph and σ be a LexBFS-ordering of G. Then, σ is a perfect elimination ordering of G, i.e., for any vertices a, b, c of G such that $a < \{b, c\}$ and $ab, ac \in E(G)$, vertices b and c must be adjacent.*

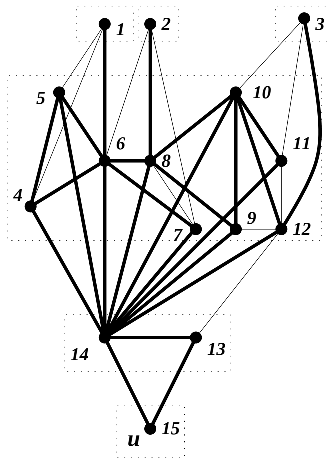

Fig. 5. An additive 3-spanner H^* of graph G presented in Figure 3

Lemma 8. *[9] Let G be an arbitrary graph and $T(G)$ be a BFS-tree of G with the root u. Let also v be a vertex of G and w ($w \neq v$) be an ancestor of v in $T(G)$ from layer $N_i(u)$. Then, for any vertex $x \in N_i(u)$ with $d_G(v, w) = d_G(v, x)$, inequality $x \leq w$ holds.*

Lemma 9. *H^* is an additive 3-spanner for G.*

Lemma 10. *If a chordal graph G has n vertices and m edges, then its additive 3-spanner H^* can be constructed in $O(m \cdot \log n)$ time.*

The main result of this subsection is the following.

Theorem 3. *Every chordal graph $G = (V, E)$ with n vertices and m edges admits an additive 3-spanner with at most $O(n \cdot \log n)$ edges. Moreover, such a sparse spanner of G can be constructed in $O(m \cdot \log n)$ time.*

In [19], it was shown that any chordal graph admits a multiplicative 3-spanner H' with at most $O(n \cdot \log n)$ edges which is constructable in $O(m \cdot \log n)$ time. It is worth to note that the spanner H' gives a better than H^* approximation of distances only for adjacent in G vertices. For pairs $x, y \in V$ at distance at least 2 in G, the stretch factor $t(x, y) = d_{H^*}(x, y)/d_G(x, y)$ given by H^* for x, y is at most 2.5 which is better than the stretch factor of 3 given by H'.

4 Spanners for *k*-Chordal Graphs

For each $l \geq 0$ define a graph Q^l with the lth sphere $N_l(u)$ as a vertex set. Two vertices $x, y \in N_l(u)$ ($l \geq 1$) are adjacent in Q^l if and only if they can be connected by a path outside the ball $B_{l-1}(u)$. Let $Q^l_1, \ldots, Q^l_{p_l}$ be all the connected components of Q^l.

Similar to chordal graphs and as shown in [8] we define a graph Γ whose vertex-set is the collection of all connected components of the graphs Q^l, $l = 0, 1, \ldots$, and two vertices are adjacent in Γ if and only if there is an edge of G between the corresponding components. The following lemma holds.

Lemma 11. *[8] Γ is a tree.*

Let u be an arbitrary vertex of a k-chordal graph $G = (V, E)$, σ be a BFS-ordering of G and T be the BFS-tree associated with σ. To construct our spanner H for G, we use the following procedure (for an example see Figure 6).

PROCEDURE 3. Additive $(k+1)$-spanners for k-chordal graphs

Input: A k-chordal graph $G = (V, E)$ with a BFS-ordering σ, and connected components Q^l_1, Q^l_2, …, $Q^l_{p_l}$ for any l, $0 \le l \le q$, where $q = max\{d_G(u, v) : v \in V\}$.

Output: A spanner $H = (V, E')$ of G.

Method:

```
E' = ∅;
for l = q downto 1 do
   for j = 1 to p_l do
      for each vertex v ∈ Q^l_j add vf(v) to E';
      pick vertex c in Q^l_j with the minimum number in σ;
      for each v ∈ Q^l_j \ {c} do
         connected = FALSE;
         while connected = FALSE do
         /* this while loop works at most ⌊k/2⌋ times for each v */
            if vc ∈ E(G) then
               add vc to E';
               connected = TRUE;
            else if vf(c) ∈ E(G) then
                  add vf(c) to E';
                  connected = TRUE;
                else v = f(v), c = f(c)
return H = (V, E').
```

Clearly, H contains all edges of BFS-tree T because for each $v \in V$ the edge $vf(v)$ is in H. For a vertex v of G, let P_v be the path of T connecting v with the root u. We call it *the maximum neighbor path of v* in G (evidently, P_v is a shortest path of G). Additionally to the edges of T, H contains also some bridging edges connecting vertices from different maximum neighbor paths.

Lemma 12. *Let c be vertex of Q^l_i with the minimum number in σ ($l \in \{1, ..., q\}$, $i \in \{1, \ldots, p_l\}$). Then, for any $a \in Q^l_i$, there is a (a, c)-path in H of length at most k consisting of a subpath $(a, \ldots, x)$ of path P_a, edge xy and a subpath $(y, \ldots, c)$ of path P_c. In particular, $d_H(a, c) \le k$. Moreover, $0 \le d_G(c, y) - d_G(a, x) \le 1$.*

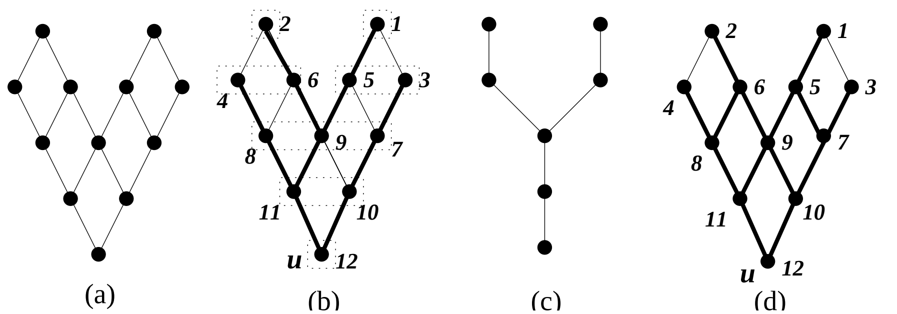

Fig. 6. (a) A 4-chordal graph G. (b) A BFS-ordering σ, BFS-tree associated with σ and a layering of G. (c) The tree Γ of G associated with that layering. (d) Additive 5-spanner (actually, additive 2-spanner) H of G constructed by PROCEDURE 3.

For any n-vertex k-chordal graph $G = (V, E)$ the following lemma holds.

Lemma 13. *H is an additive $(k+1)$-spanner of G.*

Lemma 14. *If G has n vertices, then H has at most $2n-2$ edges.*

Lemma 15. *If G is a k-chordal graph with n vertices and m edges, then H can be constructed in $O(n \cdot k + m)$ time.*

Summarizing, we have the following final result.

Theorem 4. *Every k-chordal graph $G = (V, E)$ with n vertices and m edges admits an additive $(k+1)$-spanner with at most $2n-2$ edges. Moreover, such a sparse spanner of G can be constructed in $O(n \cdot k + m)$ time.*

References

1. I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares, On sparse spanners of weighted graphs, *Discrete Comput. Geom.*, 9 (1993), 81–100.
2. H.-J. Bandelt, A. Dress, Reconstructing the shape of a tree from observed dissimilarity data, *Adv. Appl. Math.* 7 (1986) 309-343.
3. A. Brandstädt, V. Chepoi and F. Dragan. Distance Approximating Trees for Chordal and Dually Chordal Graphs, *Journal of Algorithms*, 30 (1999), 166-184.
4. A. Brandstädt, F.F. Dragan, H.-O. Le, and V.B. Le, Tree Spanners on Chordal Graphs: Complexity, Algorithms, Open Problems, to appear in the *Proceedings of ISAAC 2002.*
5. U. Brandes and D. Handke, *NP*–Completeness Results for Minimum Planar Spanners, Preprint University of Konstanz, *Konstanzer Schriften in Mathematik und Informatik,* Nr. 16, Oktober 1996.
6. L. Cai, Tree spanners: Spanning trees that approximate the distances, *Ph.D. thesis*, University of Toronto, 1992.

7. L. Cai and D.G. Corneil, Tree spanners, *SIAM J. Disc. Math.,* 8 (1995), 359–387.
8. V. Chepoi and F. Dragan, A Note on Distance Approximating Trees in Graphs, *Europ. J. Combinatorics*, 21 (2000) 761-766.
9. F.F. Dragan, Estimating All Pairs Shortest Paths in Restricted Graph Families: A Unified Approach (extended abstract), *Proc. 27th International Workshop "Graph-Theoretic Concepts in Computer Science"(WG'01),* June 2001, Springer, Lecture Notes in Computer Science 2204, pp. 103-116.
10. S.P. Fekete, J. Kremer, Tree spanners in planar graphs, *Discrete Appl. Math.* 108 (2001) 85-103.
11. J.R. Gilbert, D.J. Rose, and A. Edenbrandt, A separator theorem for chordal graphs, *SIAM J. Alg. Discrete Math.*, 5 (1984) 306-313.
12. M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, *Academic Press,* New York, 1980.
13. H.-O. Le, Effiziente Algorithmen für Baumspanner in chordalen Graphen, *Diploma thesis,* Dept. of mathematics, technical university of Berlin, 1994.
14. H.-O. Le, V.B. Le, Optimal tree 3-spanners in directed path graphs, *Networks* 34 (1999) 81-87.
15. A.L. Liestman and T. Shermer, Additive graph spanners, *Networks,* 23 (1993), 343–364.
16. M.S. Madanlal, G. Venkatesan and C.Pandu Rangan, Tree 3–spanners on interval, permutation and regular bipartite graphs, *Information Processing Letters,* 59 (1996), 97–102.
17. I.E. Papoutsakis, On the union of two tree spanners of a graph, *Preprint*, 2001.
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, *Journal of 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. D.Peleg and E.Upfal, A tradeoff between space and efficiency for routing tables. *20th ACM Symposium on the Theory of Computing,* Chicago (1988), 43-52.
22. E. Prisner, Distance approximating spanning trees, in *Proc. of STACS'97, Lecture Notes in Computer Science* 1200, (R. Reischuk and M. Morvan, eds.), Springer–Verlag, Berlin, New York, 1997, 499–510.
23. E. Prisner, H.-O. Le, H. Müller and D. Wagner, Additive tree spanners, to appear in *SIAM Journal on Discrete Mathematics.*
24. J. Soares, Graph spanners: A survey, *Congressus Numer.* 89 (1992) 225-238.