Tree Spanners in Planar Graphs
(Extended Abstract)

Sándor P. Fekete[1] and Jana Kremer[2]

[1] Center for Parallel Computing, Universität zu Köln
D–50923 Köln, GERMANY
sandor@zpr.uni-koeln.de

[2] Lehrstuhl für Volkswirtschaftslehre
Otto-Friedrich Universität Bamberg
D–96045 Bamberg
GERMANY
jana.kremer@sowi.uni-bamberg.de

Abstract. A tree t-spanner of a graph G is a spanning subtree T of G in which the distance between every pair of vertices is at most t times their distance in G. Spanner problems have received some attention, mostly in the context of communication networks. It is known that for general unweighted graphs, the problem of deciding the existence of a tree t-spanner can be solved in polynomial time for $t = 2$, while it is NP-hard for any $t \geq 4$; the case $t = 3$ is open, but has been conjectured to be hard.
In this paper, we consider tree spanners in planar graphs. We show that even for planar unweighted graphs, it is NP-hard to determine the minimum t for which a tree t-spanner exists. On the other hand, we give a polynomial algorithm for any fixed t that decides for planar unweighted graphs with bounded face length whether there is a tree t-spanner. Furthermore, we prove that it can be decided in polynomial time whether a planar unweighted graph has a tree t-spanner for $t = 3$.

1 Introduction

A t-spanner of a graph G is a spanning subgraph H of G in which the distance between every pair of vertices is at most t times their distance in G. We can think of the "stretch factor" t as the relative price increase that may incur for individual connections after replacing the network G by a cheaper subnetwork H. Spanners were first considered in the context of practical motivations from communication networks (see Peleg and Ullman [20], who introduced spanners to synchronize asynchronous networks). They have also been used for simplifying geometric data structures – see Chew [11], Dobkin, Friedman, and Supowit [12], and Arikati et al. [2]. Surveys of results on the existence and efficient constructibility can be found in [19] and [23].

Depending on the objective for choosing a subnetwork, various kinds of spanners have been considered – see the list of references for a selection of variants.

J. Hromkovič, O. Sýkora (Eds.): WG'98, LNCS 1517, pp. 298–309, 1998.

Since the main motivation is to obtain a network of small total weight, particular attention has focused on *tree spanners*, where the subnetwork H is minimal with respect to edge removal. As Cai [8], and Cai and Corneil [10] showed, the problem of deciding the existence of a tree t-spanner in an unweighted graph G can be solved in polynomial time for $t = 2$; on the other hand, the problem is NP-complete for any $t \geq 4$. The case $t = 3$ is still open, but it was conjectured in [10] to be NP-complete.

As noted above, spanners have been considered in the context of geometric distance queries – see [11,12,2]. Since planar graphs form a particularly well-understood class of sparse graphs with a number of structural and algorithmic properties that make them interesting as spanners, the focus of those works has been on *planar spanners*, where the spanning graph H is required to be planar. Also, see Brandes and Handke [7] for a proof that it is NP-hard to determine a minimum weight planar t-spanner in a graph. They also showed that determining a minimum weight t-spanner in a planar graph is an NP-hard problem.

Between considering tree spanners in general graphs and planar spanners in general graphs, it is natural to consider tree spanners in planar graphs. Not only does this allow a better understanding of the properties of graph spanners, but results on the stretch factors of tree spanners in planar graphs combine with bounds on the stretch factors of planar spanners in general graphs to yield estimates on tree spanners in general graphs.

In this paper, we show that deciding the existence of a tree t-spanner in a graph G is NP-complete, if t is part of the input, even when restricted to the situation where G is planar and unweighted. On the other hand, we prove that this problem can be solved in polynomial time for planar unweighted graphs with bounded face length and fixed t.

For some purposes, not all pairs of connections have the same importance. This motivates the concept of s,t-spanners: For a partition of $E(G)$ into two given sets of edges E_1 and E_2, a tree s,t-spanner consists of edges in E_1, and it replaces any edge $(v_1, v_2) \in E_1$ by a path of at most s times its length, and any edge $(v_1, v_2) \in E_2$ by a path of at most t times its length. We show that for fixed s and t, the existence of a tree s,t-spanner in planar unweighted graphs with bounded face length can be checked in polynomial time. By a detailed analysis of the neighborhood structures of planar graphs with tree 3-spanners, we are able to show that a planar graph has a tree 3-spanner, iff it is a subgraph of a planar graph with bounded face length that has a tree 3,12-spanner. This implies a polynomial algorithm for deciding whether a planar graph G has a tree 3-spanner.

The rest of this paper is organized as follows: In Section 2, we introduce some basic concepts. Section 3 sketches the NP-completeness of deciding the existence of a tree t-spanner in a planar graph. In Section 4, we describe the polynomial algorithm for deciding whether a planar graph with bounded face length has a tree s,t-spanner. Section 5 gives an overview of the polynomial algorithm for deciding whether a planar graph has a tree 3-spanner. In Section 6 we conclude with some open problems.

2 Preliminaries

Throughout this paper, we use the terminology of Bondy and Murty [5]. A graph G has edge set $E(G)$ and vertex set $V(G)$; we may simply write E and V when the meaning is clear. If H is a subgraph of G, then $G - H$ denotes the graph obtained by deleting from G all edges of H. For a pair of vertices v_1 and v_2 in a connected graph G, we denote the length of a shortest path from v_1 to v_2 by $d_G(v_1, v_2)$. We will concentrate on the case of unweighted graphs without loops, so for any edge $(v_1, v_2) \in E(G)$, we have $d_G(v_1, v_2) = 1$. For a planar graph G, we write G^* for the dual graph. For $S \subset V$, the number of the edges leaving S in the graph G is denoted by $\delta_G(S)$. For $S \subset V$, we denote by $N(S)$ the set of neighbors of S, i. e., the set of vertices $v \in V \setminus S$ with a $w \in S$, such that $(v, w) \in E$. For a set of vertices $S \subseteq V$, the subgraph induced by S is denoted by $G[S]$.

For a real number $t \geq 1$, a subgraph H of a connected graph G is a *t-spanner* if $d_H(v_1, v_2) \leq t \cdot d_G(v_1, v_2)$ for all $v_1, v_2 \in E(G)$. A *tree t-spanner* is a t-spanner that is a tree. The parameter t is called the *stretch factor*; the smallest value t for which a graph G has a tree t-spanner is called the *tree stretch index* of G, denoted by $\sigma_T(G)$. It was shown in [10] that the following holds:

Lemma 1 *A subgraph H of a connected graph G is a t-spanner, iff for all edges $(v_1, v_2) \in E(G) - E(H)$, we have $d_H(v_1, v_2) \leq t$*

This allows us to consider only integer stretch factors for unweighted graphs. If the condition $d_H(v_1, v_2) \leq t$ is satisfied for a particular edge $e = (v_1, v_2) \in E(G) - E(H)$, we say that e has a *short detour* in H; for the case of tree spanners T, there is a unique corresponding shortest path, denoted by $p_T(e)$.

3 An NP-Completeness Result

It was shown in [10] that it is NP-complete to decide whether $\sigma_T(G) \leq t$ for a general unweighted graph, as long as $t \geq 4$. In this section, we sketch our proof that it is NP-complete to decide $\sigma_T(G) \leq t$ for a planar unweighted graph, where t is part of the input. Our reduction is from a special subclass of 3-SAT instances, called Planar 3SAT, which was shown to be NP-complete by Lichtenstein [16].

A 3SAT instance I is said to be an instance of Planar 3SAT, if the following bipartite graph G_I is planar: Every variable and every clause in I is represented by a vertex in G_I; two vertices are connected, if and only if one of them represents a variable that appears in the clause that is represented by the other vertex. See Figure 1 (a) for an example.

In the following, we sketch the necessary gadgets for our hardness proof. Details are contained in the full version of the paper, see also [15].

3.1 The Basic Setup

In a first step, the graph G_I is transformed into a graph G'_I. As shown in Figure 1, each set of three edges adjacent to the same clause vertex is replaced by three

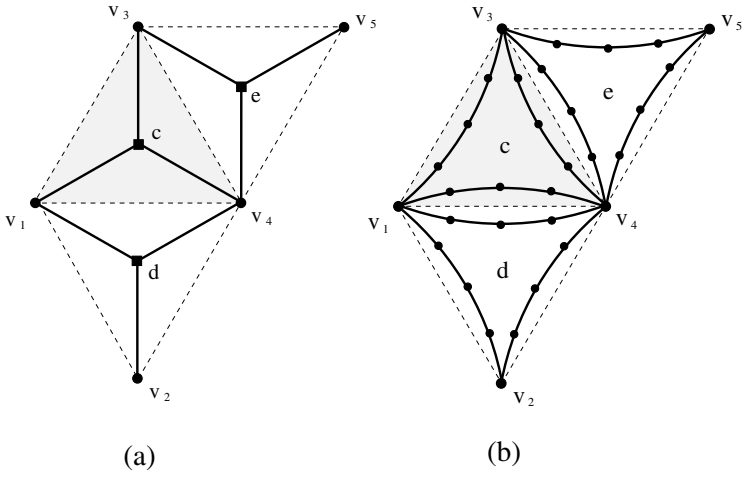

Fig. 1. (a) The graph G_I representing the PLANAR 3SAT instance $(x_1 \vee x_3 \vee x_4) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_4) \wedge (\bar{x}_3 \vee x_4 \vee \bar{x}_5)$; (b) The transformed graph G'_I

paths of length 4. From this graph G'_I, *any* spanning tree T' is chosen. This spanning tree has a certain stretch factor t', which is polynomially bounded by the size of I.

For the second step, we add edges and vertices to G'_I to get a graph G''_I. In particular, we use the gadgets shown in Figure 2 to make sure that for $t = t' + 1$, all the edges of T' must be contained in a potential tree t-spanner of G''_I, if there is one.

The gadget shown in (a) has been used extensively in the proofs of [10] and [7]. It is easy to see that any tree 5-spanner of the graph G shown in the figure must contain the edge e. In the following, edges *forced* in this way are indicated by bold drawing.

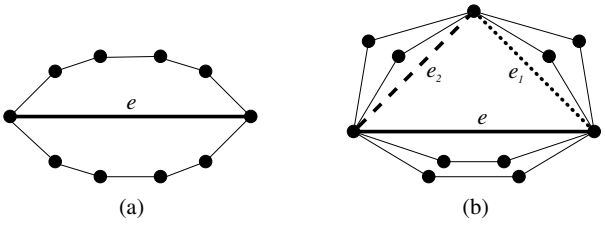

Fig. 2. (a) A forced edge; (b) a forced pair

Figure 2 (b) shows another gadget that can be used for forcing one out of two edges: Any tree 3-spanner must contain e and precisely one of the two edges e_1 and e_2.

In a third step, components for clauses and variables are added to G''_I. The following two subsections give a rough description of their design and properties.

3.2 Gadgets for Variables

Figure 3 shows the gadget G_{var} for representing variables. It consists of a central "variable" vertex v, connected to "literal" vertices v_1, $\overline{v_1}$, ..., v_s, $\overline{v_s}$. v_i and $\overline{v_i}$ are connected by an edge $w^{(i)}$ that is forced by two paths of length t. $\overline{v_i}$ and v_{i+1} (indices modulo s) are connected by a path of length $t-2$, containing the vertices $\overline{v_i}, w_1^{(i,i+1)}, \ldots, w_{t-3}^{(i,i+1)}, v_{i+1}$. The edge $f_i = (\overline{v_i}, w_1^{(i,i+1)})$ is not forced, all other edges of the path are. Connections to the outside, i. e., to the rest of the graph, are at the literal vertices.

Furthermore, no two literal vertices are adjacent and there is no outside vertex that is connected for all $1 \leq i \leq s$ to at least one af the vertices $v_i, \bar{v}_i$.

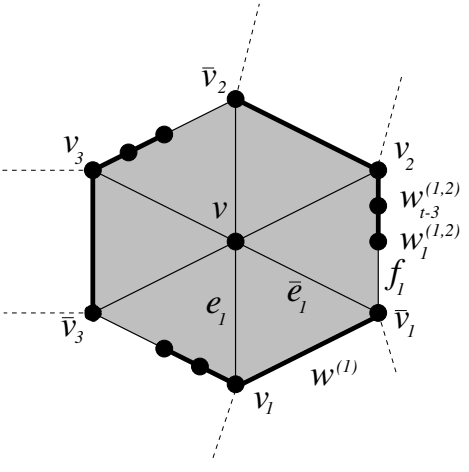

Fig. 3. Variable component G_{var}

Using straightforward induction, it is not hard to prove the following:

Lemma 2 *A tree t-spanner of a graph containing G_{var} cannot contain any of the edges f_i and must contain precisely one of the edges e_i, $\overline{e_i}$. Furthermore, e_1 is contained iff all e_i are contained.*

Containment of e_i or $\overline{e_i}$ corresponds to a truth assignment of the represented variable.

3.3 Gadgets for Clauses

Due to space limitations, we cannot give full technical details of the clause gadgets, but the basic idea is shown in Figure 4 (a). Figure 4 (b) shows the general layout for combining clauses and variables.

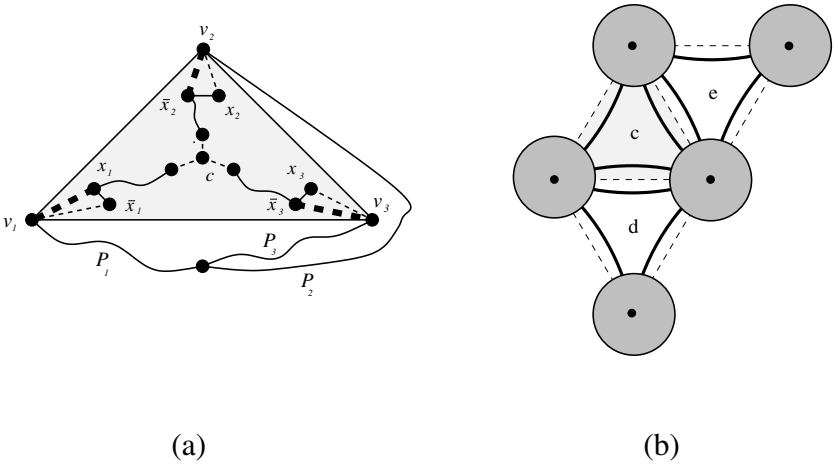

Fig. 4. (a) Idea of the clause component, shown for $(x_1 \vee \overline{x_2} \vee \overline{x_3})$; (b) combination of clause gadgets (triangles) with variable gadgets (circles)

Around a central node c, we group three forced paths of appropriate length, starting with edges (c, u_1), (c, u_2), (c, u_3). These paths connect to literal nodes of the corresponding variables. The choice of path lengths, forced edges, forced pairs and connections to variable components is done in a way that forces c to be a leaf of a tree t-spanner, if there is one. Furthermore, the existence of a tree t-spanner hinges on the existence of short detours $p_T(c, u_i)$, $p_T(c, u_j)$ for the two edges (c, u_i), (c, u_j) adjacent to c that are not contained in a spanner T.

Each non-true literal forces an extra edge into a potential short detour $p_T(c, u_i)$. The path lengths are set up in a way that allows one extra edge, but not two of them. This forces at least one satisfying literal to be in each clause. Conversely, if there is a truth assignment, we can keep c connected to the path that leads directly to the satisfying literal, making sure that there can be at most one extra edge for the detours $p_T(c, u_i)$, $p_T(c, u_j)$.

We summarize:

Theorem 3. *It is NP-complete to decide $\sigma_T(G) \leq t$ for planar unweighted graphs G and integers t.*

4 Planar Graphs with Bounded Face Length

In this section, we show that deciding the existence of a tree t-spanner in a planar graph with all faces of bounded length can be performed in polynomial time.

For this purpose, we introduce the notion of a *c-cut tree* in a graph:

Definition 4 *Let T be a spanning tree in a graph G. Removing any edge $e \in T$ splits T into two connected components, inducing a partition of the vertex set into $P_T(e) = (V_T(e), V'_T(e))$. We say that T is a c-cut tree in G, if for all $e \in T$, $|\delta_G(V_T(e))| \leq c$.*

It is straightforward to show that the following holds:

Lemma 5 *A planar graph G has a tree t-spanner, iff G^* has a $(t+1)$-cut tree.*

Furthermore, we can establish the following constructive characterization of c-cut trees:

Lemma 6 *A planar graph G has a c-cut tree, iff there is a "rooted nested family" $F \subseteq 2^V \times V$ with the following properties:*

1. *$(V, r) \in F$ for an $r \in V$*
2. *$r \in S$ for all $(S, r) \in F$,*
3. *$|\delta_G(S)| \leq c$ for all $(S, r) \in F$,*
4. *for all (S_1, r_1), $(S_2, r_2) \in F$ we have $S_1 \subseteq S_2$ or $S_1 \subseteq V \setminus S_2$,*
5. *for all $(S, r) \in F$ there are $l \geq 1$ and $(S_i, r_i) \in F$, $1 \leq i \leq l$, with $S \setminus \{r\} = \dot{\cup} S_i$ and $(r, r_i) \in E$.*

The vertex sets S correspond to the subsets of a partition induced by the removal of an edge $e \in T$ from T, while $r \in S$ is the vertex adjacent to e. The proof is straightforward and omitted.

Using the characterization from Lemma 6, we get the following result:

Theorem 7. *For fixed t, it can be decided in polynomial time for planar unweighted graphs G with bounded face length whether $\sigma_T(G) \leq t$.*

Sketch: Consider the existence of a rooted nested family F of G^* as described in Lemma 6. Since t is fixed, there are only polynomially many possible cuts in G^* of size not larger than $t+1$, implying we only have to consider polynomially many sets (S, r) that can be used for F. Since all faces in G have bounded length, the dual graph G^* has bounded degree, so there is a polynomial number of possible partitions for any (S, r). Using dynamic programming and proceeding by increasing size of S, we can decide the existence of a rooted nested family as described in Lemma 6 in polynomial time.

□

As described in the introduction, the concept of tree t-spanners can be generalized:

Definition 8 *Let G be a graph with $E(G) = E_1 \dot\cup E_2$. Then a spanning tree T of G is a tree s,t-spanner for $G = (V, E_1 \dot\cup E_2)$, iff T is a subgraph of (V, E_1), and for all edges $(v_1, v_2) \in E_1 - T$, we have $d_T(v_1, v_2) \leq s$, and for all edges $(v_1, v_2) \in E_2 - T$, we have $d_T(v_1, v_2) \leq t$.*

With an analogous approach to the one for tree t-spanners, we can establish the following result for tree s,t-spanners:

Theorem 9. *For fixed s and t, it can be decided in polynomial time for planar unweighted graphs $G = (V, E_1 \dot\cup E_2)$ of bounded face length, possibly with multi-edges, whether there is a tree s,t-spanner.*

5 Deciding the Existence of Tree 3-Spanners in Planar Graphs

In this section, we sketch the polynomial algorithm for deciding whether a planar unweighted graph G has a 3-spanner. The key idea is to add a set of edges E' to obtain a graph $G_{\leq 4}$ with face length bounded by 4, such that $G_{\leq 4} = (V, E \dot\cup E')$ has a tree $3,12$-spanner, iff G has a tree 3-spanner. The existence of a $3,12$-spanner in $G_{\leq 4}$ can be decided in polynomial time by the algorithm from the previous section.

If there is no face of length more than 4, we are done, so consider a face bounded by the chordless cycle $C = v_1, \ldots, v_s$ with $|C| \geq 5$. Now assume there is a tree 3-spanner in G. Since $|C| \geq 5$, for any edge $e = (v_i, v_{i+1})$ in $C - T$, there must be a path in T that is not longer than 3 and not fully contained in C. The different possibilities for such a path are shown in Figure 5.

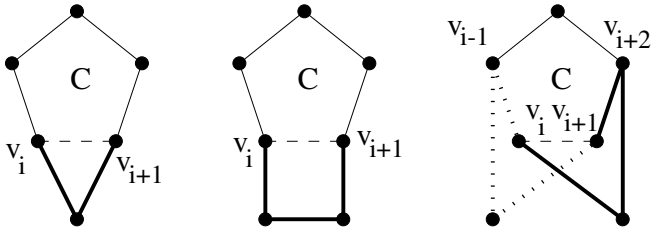

Fig. 5. Different possibilities for a short detour $p_T(v_i, v_{i+1})$ of an edge (v_i, v_{i+1}) in $C - T$

Now we can analyze the structure of T in the neighborhood of C: consider the edges in $p_T(v_i, v_{i+1}) - C$. It is not hard to see that each of these edges must be contained in $p_T(v_j, v_{j+1})$ for an edge $(v_j, v_{j+1}) \in ((C - T) - (v_i, v_{i+1}))$, since T cannot contain a cycle. Because $p_T(v_j, v_{j+1})$ contains at most three edges, (v_j, v_{j+1}) is adjacent to (v_i, v_{i+1}) or both are adjacent to the same edge in C.

From this, we can derive the following lemma:

Lemma 10 *Let G be a planar graph with a tree 3-spanner T. If C is a chordless cycle in G, $|C| \geq 5$, then there is a "semi-dominating" tree T_C in T, such that*

1. *C is "weakly dominated" by $V(T_C)$, i. e., for any vertex $v_i \in C$, v_i is adjacent to T_C, or both its neighbors v_{i-1} and v_{i+1} are.*
2. *If a vertex $v \in C$ is not adjacent to T_C, then both of its neighbors are adjacent to the same vertex of T_C.*

Furthermore, for a given cycle C that bounds a face F of G, the semi-dominating tree is uniquely determined.

An example is shown in Figure 6. Bold lines show the semi-dominating tree.

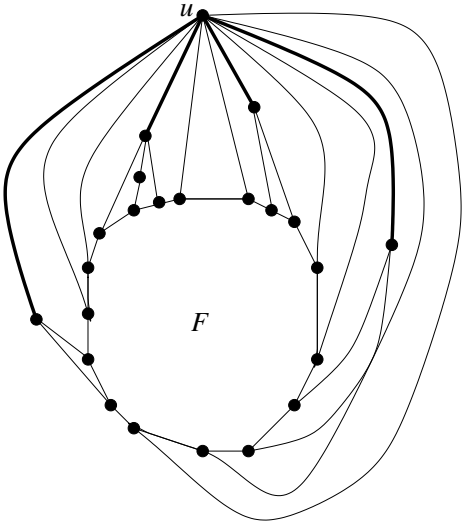

Fig. 6. A long chordless cycle C in G and a semi-dominating tree T_C

Now consider a vertex $u \in T_C$ as shown in Figure 7. If u does not weakly dominate C (which would imply $u = T$), then it induces a subdivision (called the *u-subdivision*) of C as follows. Let D_i be a maximal path weakly dominated by u. The first vertex of D_i is denoted by d_i^h, the last by d_i^t. Between any two D_i and D_{i+1}, there is a path P_i, consisting of vertices that are non-adjacent to u. Clearly, any P_i must contain at least two vertices. For any i, let P_i^1 be the path (d_i^t, P_i, d_{i+1}^h), while P_i^2 is the path $(D_{i+1}, P_{i+1}, \ldots, P_{i-1}, D_i)$.

Now we insert a set $E'(u)$ of new edges as follows. For any i, insert the edge (d_i^t, d_{i+1}^h) – shown by broken lines in Figure 7. This yields a face $F(u)$ that is dominated by u. This face is then triangulated by further new edges. Using the structure of the semi-dominating tree, we can show that the face bounded by C is subdivided into faces of length at most 4. Furthermore, the end vertices of the

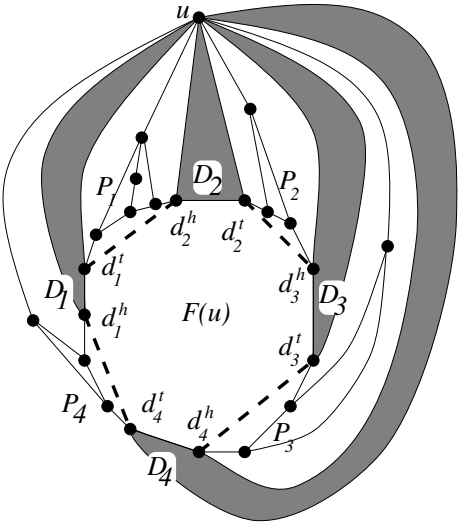

Fig. 7. A vertex u in a semi-dominating tree T_C, its u-partition, and the subface $F(u)$

new edges are connected by paths with at most 12 edges in a tree 3-spanner T of G. Note that in the process of introducing new edges, we may create multi-edges.

After inserting a new chord for any chord of a face, this subdivision is carried out for every face that is bounded by a chordless cycle C with more than four edges and for all vertices of the semi-dominating tree T_C of C. Eventually we get the planar supergraph $G_{\leq 4}$ with the desired properties.

Conversely, any tree $3, 12$-spanner in the expanded graph $(V, E(G) \dot\cup E')$ induces a tree 3-spanner in G. (Full details can be found in [15].)

The following definition and Lemma 12 show how to find a semi-dominating tree of a cycle in polynomial time. Once the semi-dominating trees are found, the procedure of inserting the edges, and testing for the existence of a tree $3, 12$-spanner yields a polynomial algorithm – recall Theorem 9.

Definition 11 *Let $u \in N(C)$ be a vertex that does not weakly dominate the cycle C. Let $D_1, P_1, \ldots, D_r, P_r$ be the u-subdivision of C.*

A vertex $w \in N(C)$ is an independent C-neighbor *of u, if it is adjacent to u in G and if there is an index $1 \leq i \leq r$ such that the following conditions hold:*

1. *There is a path of at most two edges in G that connects w with a vertex of P_i, and*
2. *there are vertices w_i^h, w_i^t from P_i^1 that are adjacent to w in G and vertices u_i^h, u_i^t from P_i^2 that are adjacent to u in G, such that w_i^h, w_i^t, u_i^h, and u_i^t are pairwise disjoint and $u_i^h w_i^t$, $w_i^h u_i^t \in E(C)$ holds.*

(Note that the path does not contain vertex u, since u is not adjacent to any vertex in P_i.)

The set of all independent C-neighbors is denoted by $N(C, u)$. A vertex $w \in N(C)$ is a C-successor *of u, if there is a path $(w_0, w_1, \ldots, w_k$ with $w_0 = u$, $w_k = w$, such that for any $1 \leq i \leq k$, the vertex w_i is an independent C-neighbor of w_{i-1}. The set of all C-successors is denoted by $D(C, u)$.*

Lemma 12 *Let C be a cycle in a planar graph G, and let u be adjacent to a vertex in C.*

If C has a semi-dominating tree T_C containing u, then

$$T_C = G[D(C, u)] - \{(v, w) : w \notin N(C, v)\}.$$

Summarizing, we get

Theorem 13. *We can decide in polynomial time whether a planar unweighted graph G has a tree 3-spanner.*

6 Conclusion

In this paper, we have shown that for planar graphs, it is possible to decide the existence of a tree 3-spanner in polynomial time. Our method makes strong use of planarity, yet the resulting algorithm is rather complicated. It has been conjectured that deciding the existence of a tree 3-spanner is an NP-complete problem, and our impression from the experience with planar graphs seems to support this belief.

On the other hand, we could prove that deciding the existence of a tree t-spanner is NP-complete, as long as t is part of the input. The complexity for fixed t is unclear, but there may be a polynomial method of deciding the question, possibly using a combination of dynamic programming and an analysis of neighborhood structures, as we did for the case $t = 3$. Unfortunately, this analysis appears to become rather tedious even for $t = 4$.

Acknowledgment

We would like to thank Dorothea Wagner, Ulrik Brandes, and Dagmar Handke for helpful discussions.

References

1. I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete and Computational Geometry*, **9** (1993), pp. 81–100.
2. S. Arikati, D. Z. Chen, L. P. Chew, G. Das, M. Smid, and D. Zaroliagis. Planar spanners and approximate shortest path queries among obstacles in the plane.. In: J. Diaz, ed., *Algorithms - ESA '96*, Springer Lecture Notes in Computer Science #1136, 1996, pp. 514–528.

3. B. Awerbuch, A. Baratz, and D. Peleg. Efficient broadcast and light-weight spanners. Manuscript, 1992.
4. S. Bhatt, F. Chung, F. Leighton, and A. Rosenberg. Optimal simulations of tree machines. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science* (FOCS 1986), pp. 274-282.
5. J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. North-Holland, New York, 1976.
6. K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, **13** (1976), pp. 335–379.
7. U. Brandes and D. Handke. NP-completeness results for minimum planar spanners. In *Proceedings of the 23th Workshop on Graph-Theoretic Concepts in Computer Science* (WG '97), Springer Lecture Notes in Computer Science # 1335, 1997, pp. 85-99.
8. L. Cai. Tree spanners: spanning trees that approximate distances. Ph. D. thesis, University of Toronto, Toronto, Canada, 1992. Available as *Technical Report 260/92*, Department of Computer Science, University of Toronto.
9. L. Cai. NP-completeness of minimum spanner problems, *Discrete Applied Mathematics*, **48** (1994), pp. 187–194.
10. L. Cai and D. G. Corneil. Tree spanners. *SIAM Journal of Discrete Mathematics*, **8** (1995), pp. 359-387.
11. L. P. Chew. There is a planar graph almost as good as the complete graph. In *Proceedings of the 2nd ACM Symposium on Computational Geometry*, pp. 169-177, 1986.
12. D. P. Dobkin, S. J. Friedman, and K. J. Supowit. Delaunay graphs are almost as good as complete graphs. *Discrete and Computational Geometry*, **5** (1990), pp. 399–407.
13. J. E. Hopcroft and R. E. Tarjan. Efficient planarity testing. *Journal of the ACM*, **21**, (1974), 549–568.
14. G. Kortsarz and D. Peleg. Generating sparse 2-spanners. In: *Proceedings of the Third Scandinavian Workshop on Algorithm Theory* (SWAT 1992).
15. J. Kremer. Baumspanner in planaren Graphen. *Diploma thesis*, Mathematisches Institut, Universität zu Köln, 1997.
16. D. Lichtenstein. Planar formulae and their uses. *SIAM Journal of Computing*, **11** (1982), pp. 329–343.
17. A. L. Liestman and T. Shermer. Grid spanners. *Networks*, **23** (1993), pp. 123–133.
18. A. Mansfield. Determining the thickness of graphs is NP-hard. *Mathematical Proceedings of the Cambridge Philosophical Society*, **93** (1983), pp. 9–23.
19. D. Peleg and A. A. Schäffer. Graph spanners. *Journal of Graph Theory*, **13** (1989), pp. 99–116.
20. D. Peleg and J. D. Ullman. An optimal synchronizer for the hypercube. In *Proceedings of the 6th ACM Symposium on Principles of Distributed Computing*, 1987, pp. 77–85.
21. 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*, (STOC 1988), pp. 43–52.
22. D. Richards and A. L. Liestman. Degree-constrained pyramid spanners. *Parallel and Distributed Computing*, **25** (1995), pp. 1–6.
23. J. Soares. Graph spanners: a survey. *Congressus Numerantium*, **89** (1992),pp. 225–238.