SIAM J. COMPUT.
Vol. 25, No. 2, pp. 355–368, April 1996

008

LOW-DEGREE SPANNING TREES OF SMALL WEIGHT*

SAMIR KHULLER†, BALAJI RAGHAVACHARI‡, AND NEAL YOUNG§

Abstract. Given n points in the plane, the degree-K spanning-tree problem asks for a spanning tree of minimum weight in which the degree of each vertex is at most K. This paper addresses the problem of computing low-weight degree-K spanning trees for $K > 2$. It is shown that for an arbitrary collection of n points in the plane, there exists a spanning tree of degree 3 whose weight is at most 1.5 times the weight of a minimum spanning tree. It is shown that there exists a spanning tree of degree 4 whose weight is at most 1.25 times the weight of a minimum spanning tree. These results solve open problems posed by Papadimitriou and Vazirani. Moreover, if a minimum spanning tree is given as part of the input, the trees can be computed in $O(n)$ time.

The results are generalized to points in higher dimensions. It is shown that for any $d \geq 3$, an arbitrary collection of points in $\Re^d$ contains a spanning tree of degree 3 whose weight is at most 5/3 times the weight of a minimum spanning tree. This is the first paper that achieves factors better than 2 for these problems.

Key words. algorithms, graphs, spanning trees, approximation algorithms, geometry

AMS subject classifications. 05C05, 05C10, 05C85, 65Y25, 68Q20, 68R10, 68U05, 90C27, 90C35

1. Introduction.

Given n points in the plane, how do we find a spanning tree of minimum weight among those in which each vertex has degree at most K? Here the weight of an edge between two points is defined to be the Euclidean distance between them. This problem is referred to as the *Euclidean degree-K spanning tree problem* and is a generalization of the Hamilton path problem, which is known to be NP-hard [10, 12]. When $K = 3$, it was shown to be NP-hard by Papadimitriou and Vazirani [15], who conjectured that it is NP-hard for $K = 4$ as well. When $K = 5$, the problem can be solved in polynomial time [14].

This paper addresses the problem of computing low-weight degree-K spanning trees for $K > 2$. In any metric space, it is known that there always exists a spanning tree of degree 2 whose cost is at most twice the cost of a minimum spanning tree (MST). This is shown by taking a Euler tour of an MST (in which each edge is taken twice) and producing a Hamilton tour by short-cutting the Euler tour. In the case of general metric spaces, it is easy to generate examples in which the ratio of a shortest Hamilton path to the weight of an MST is arbitrarily close to 2. But such examples do not translate to points in $\Re^d$. In view of this, Papadimitriou and Vazirani [15] posed the problem of obtaining factors better than 2 for the Euclidean degree-K spanning-tree problem. It should be noted that in the special case of $K = 2$, Christofides [3] gave a simple and elegant polynomial-time approximation algorithm with an approximation ratio of 1.5 for computing a traveling salesperson tour for points satisfying the triangle inequality (points in a metric space).

---

* Received by the editors February 3, 1994; accepted for publication (in revised form) August 31, 1994.

† Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (samir@cs.umd.edu). The research of this author was supported by NSF Research Initiation Award CCR-9307462.

‡ Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083 (rbk@utdallas.edu). The research of this author was supported by NSF Research Initiation Award CCR-9409625.

§ Department of Computer Science, Dartmouth College, Hanover, NH 03755-3510 (neal.young@dartmouth.edu). This research was done while the author was at Cornell University and at UMIACS and was supported in part by NSF grants CCR-8906949 and CCR-9111348.

1.1. Our contributions. In this paper, we show that for an arbitrary collection of n points in the plane, there exists a degree-3 spanning tree whose weight is at most 1.5 times the weight of an MST. We also show that there exists a degree-4 spanning tree whose weight is at most 1.25 times the weight of an MST. This solves an open problem posed by Papadimitriou and Vazirani [15].

Moreover, if an MST is given as part of the input, the trees can be computed in $O(n)$ time. Note that our bound of 1.5 for the degree-3 spanning-tree problem is an "absolute" guarantee (based on the weight of an MST) as opposed to a "relative" guarantee for the degree-2 spanning tree obtained by Christofides [3] (based on the weight of an optimal solution).

We also generalize our results to points in higher dimensions. We show that for any $d \geq 2$, an arbitrary collection of points in $\Re^d$ contains a degree-3 spanning tree whose weight is at most 5/3 times the weight of an MST. This is the first paper that achieves factors better than 2 for these problems.

1.2. Significance of our results. Many approximation algorithms make use of the triangle inequality to obtain approximate solutions to NP-hard problems. These algorithms typically involve a "short-cutting" step where the triangle inequality is used to bound the cost of the obtained solution. Examples include Christofides's heuristic for the traveling salesperson problem [3], biconnectivity augmentation [8], approximate weighted matching [11], prize-collecting traveling salesperson [2], and bounded-degree subgraphs which have low weight and small bottleneck cost [16].

A question of general interest is how to obtain improved approximation algorithms for such problems when the points come from a Euclidean, as opposed to arbitrary, metric space. This requires making use of more than just the triangle inequality. Surprisingly, for most problems, improved algorithms are not known. (A notable exception is the famous Euclidean Steiner tree problem [5, 6].) We use rudimentary geometric techniques to obtain an improved algorithm for the Euclidean degree-K spanning-tree problem.

The key to our method is to give short-cutting steps that are provably better than implied by the triangle inequality alone. Lemma 3.3, which bounds the perimeter of an arbitrary triangle in terms of distances to its vertices from any point, is typical of the techniques that we use to get better bounds.

1.3. Related work. Papadimitriou and Vazirani showed that any MST whose vertices have integer coordinates has maximum degree at most 5 [15]. Monma and Suri [14] showed that for *every* set of points in the plane, there exists a degree-5 MST.

Many recent works have given algorithms to find subgraphs of bounded degree that simultaneously satisfy other given constraints. A polynomial-time algorithm to find a spanning tree or a Steiner tree of a given subset of vertices in a graph with degree at most one more than minimum was given by Fürer and Raghavachari [9]. This was extended to weighted graphs by Fischer [7]. He showed how to find MSTs whose degree is within a constant multiplicative factor plus an additive $O(\log n)$ of the optimal degree. The degree bound is improved further in the case when the number of different edge weights is bounded by a constant. Ravi et al. [16] consider the problem of computing bounded-degree subgraphs satisfying given connectivity properties in a graph whose edge weights satisfy the triangle inequality. They give efficient algorithms for computing subgraphs which have low weight and small bottleneck cost. Salowe [18] and Das and Heffernan [4] consider the problem of computing bounded-degree graph spanners and provide algorithms for computing them. Robins and Salowe [17] study the maximum degrees of MSTs under various metrics.

2. Preliminaries. Let $V = \{v_1, \ldots, v_n\}$ be a set of n points in the plane. Let G be the complete graph induced by V, where the weight of an edge is the Euclidean distance between its endpoints. We use the terms points and vertices interchangeably. Let $\overline{uv}$ be the Euclidean distance between vertices u and v. Let $T_{\min}$ be an MST of the points in V. Let $w(T)$ denote the total weight of a spanning tree T. Let T_k denote a spanning tree in which every vertex has degree at most k. Let $\deg_T(v)$ be the degree of a vertex v in the tree T. Let ΔABC denote the triangle formed by points A, B, and C. Let $\angle ABC$ denote the angle formed at B between line segments AB and BC. Let $\overline{ABC}$ denote the perimeter of ΔABC; and more generally, let $\overline{v_1 v_2 \ldots v_k}$ denote the perimeter of the polygon formed by the line segments $v_i v_{i+1}$ for $1 \leq i \leq k$, where $v_{k+1} = v_1$.

In this paper, we prove the following: for an arbitrary set of points in $\Re^2$,

$$\exists T_3 : \quad w(T_3) \leq 1.5 \times w(T_{\min}), \tag{1}$$

$$\exists T_4 : \quad w(T_4) \leq 1.25 \times w(T_{\min}). \tag{2}$$

For an arbitrary set of points in $\Re^d$ $(d > 2)$,

$$\exists T_3 : \quad w(T_3) \leq \frac{5}{3} \times w(T_{\min}). \tag{3}$$

3. Points in the plane. We first consider the case of $\Re^2$—points in the plane. We first note some useful properties of MSTs in $\Re^d$.

PROPOSITION 3.1 ([15]). *Let AB and BC be two edges incident to a point B in an MST of a set of points in $\Re^d$. Then $\angle ABC$ is a largest angle in ΔABC.*

COROLLARY 3.2. *Let AB and BC be two edges incident to a point B in an MST of a set of points in $\Re^d$. Then*
- $\angle ABC \geq 60°$,
- $\angle BAC, \angle BCA \leq 90°$.

3.1. An upper bound on the perimeter of a triangle. We now prove an upper bound on the perimeter of an arbitrary triangle in terms of distances to its vertices from an arbitrary point. This lemma is useful in proving the performances of our algorithms. The lemma is also interesting in its own right, and we believe that it and the associated techniques will be useful in other geometrical problems.

LEMMA 3.3. *Let X, A, B, and C be points in $\Re^d$ with $\overline{XA} \leq \overline{XB}, \overline{XC}$. Then*

$$\overline{ABC} \leq (3\sqrt{3} - 4)\overline{XA} + 2(\overline{XB} + \overline{XC}). \tag{4}$$

Note that $3\sqrt{3} - 4 \approx 1.2$. Recall that $\overline{ABC}$ is the perimeter of the triangle and $\overline{XY}$ is the distance from X to Y.

Proof. Let B' and C' be points on XB and XC, respectively, such that $\overline{XA} = \overline{XB'} = \overline{XC'}$ (see Fig. 1). First we observe that the lemma is true if it is true for the points X, A, B', and C'. This follows because by the triangle inequality,

$$\overline{ABC} \leq \overline{AB'C'} + 2\overline{BB'} + 2\overline{CC'}.$$

By our assumption,

$$\overline{AB'C'} \leq (3\sqrt{3} - 4)\overline{XA} + 2(\overline{XB'} + \overline{XC'}).$$

Combining the two inequalities yields the desired result. Therefore, in the rest of the proof, we show that the lemma is true when the "arms" $\overline{XA}$, $\overline{XB'}$, and $\overline{XC'}$ are equal.

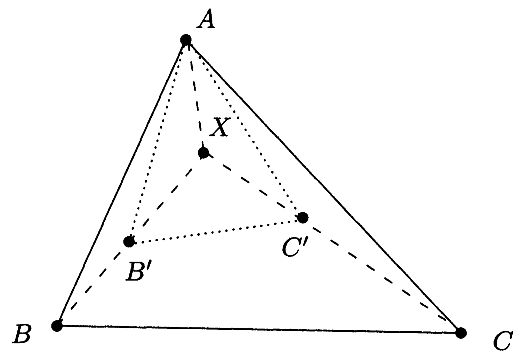

FIG. 1. *Shrinking to obtain canonical form.*

It is not very difficult to see that to maximize the perimeter of the triangle, X will be in the plane defined by A, B', and C', and thus X is at the center of a circle passing through A, B', and C'.

By scaling, it suffices to consider the case when the circle has unit radius. In this case, the right-hand side (r.h.s.) of (4) is exactly $3\sqrt{3}$. Thus, it suffices to show that the maximum perimeter achieved by any triangle whose vertices lie on a unit circle is $3\sqrt{3}$. This is easily proved [13]. $\square$

Note that in an arbitrary metric space it is possible to have an (equilateral) triangle of perimeter six and a point X at distance one from each vertex.

3.2. Spanning trees of degree 3. We now assume that we are given a Euclidean MST T of degree at most 5. We show how to convert T into a tree of degree at most 3. The weight of the resulting tree is at most 1.5 times the weight of T.

High level description. The tree T is rooted at an arbitrary leaf vertex. Since T is a degree-5 tree, once it is rooted at a leaf, each vertex has at most four children. For each vertex v, the shortest path P_v starting at v and visiting every child of v is computed. The final tree T_3 consists of the union of the paths $\{P_v\}$. Figure 2 gives the above algorithm. In analyzing the algorithm, we think of each vertex v as replacing its edges from its children with the path P_v. The above technique of "short-cutting" the children of a vertex by "stringing" them together has been known before, especially in the context of computing degree-3 trees in metric spaces (see [16, 18]).

TREE-3(V, T) — *Find a degree-3 tree of V.*

```
Root the MST T at a leaf vertex r.
For each vertex v ∈ V do
    Compute P_v, the shortest path starting at v and visiting all the children of v.
Return T_3, the tree formed by the union of the paths {P_v}.
```

FIG. 2. *Algorithm to find a degree-3 tree.*

Note. Typically, the initial MST has very few nodes with degree greater than 3 [1]. In practice, it is worth modifying the algorithm to scan the vertices in preorder, maintaining the partial tree T_3 of edges added so far, and to add paths to T_3 as follows. When considering a vertex v, if the degree of v in the partial T_3 is 2, add the path P_v as described in the algorithm. Otherwise, its degree is 1, so, in this case, relax the requirement that the added path must start at v. That is, add the shortest path that visits v and all of v's children to T_3 (see §3.3). This modification will never increase the cost of the resulting tree but may offer substantially lighter trees in practice.

LEMMA 3.4. *The algorithm in Fig.* 2 *outputs a spanning tree of degree* 3.

Proof. An easy proof by induction shows that the union of the paths forms a tree. Each vertex v is on at most two paths and is an interior vertex of at most one path. $\square$

LEMMA 3.5. *Let v be a vertex in an MST T of a set of points in $\Re^2$. Let P_v be a shortest path visiting $\{v\} \cup child_T(v)$ with v as one of its endpoints.*

$$w(P_v) \leq 1.5 \times \sum_{v_i \in child_T(v)} \overline{vv_i}.$$

By the above lemma, each path P_v has weight at most 1.5 times the weight of the edges it replaces. Thus we have the following theorem.

THEOREM 3.6. *Let T be an MST of a set of points in $\Re^2$. Let T_3 be the spanning tree output by the algorithm in Fig.* 2.

$$w(T_3) \leq 1.5 \times w(T).$$

Proof of Lemma 3.5. We consider the various cases that arise depending on the number of children of v. The cases when v has no children or exactly one child are trivial.

Case 1. *v has* 2 *children, v_1, v_2.* There are two possible paths for P_v, namely $P_1 = [v, v_1, v_2]$ and $P_2 = [v, v_2, v_1]$. Clearly,

$$w(P_v) = \min(w(P_1), w(P_2)) \leq \frac{w(P_1) + w(P_2)}{2} = \frac{\overline{vv_1}}{2} + \frac{\overline{vv_2}}{2} + \overline{v_1v_2} \leq 1.5\ (\overline{vv_1} + \overline{vv_2}).$$

Case 2. *v has* 3 *children, v_1, v_2, v_3.* Let v_1 be the child that is nearest to v. Consider the following four paths (see Fig. 3): $P_1 = [v, v_1, v_2, v_3], P_2 = [v, v_1, v_3, v_2], P_3 = [v, v_2, v_1, v_3]$, and $P_4 = [v, v_3, v_1, v_2]$.

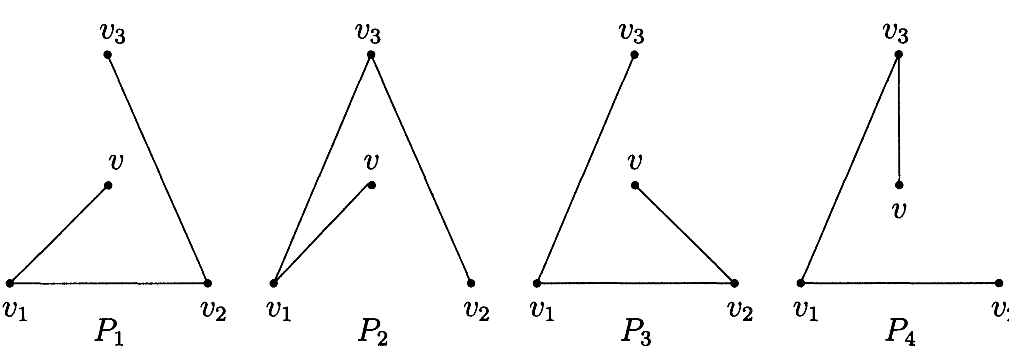

FIG. 3. *T_3, three children.*

The path P_v is at most as heavy as the lightest of $\{P_1, P_2, P_3, P_4\}$. The weight of the lightest of these paths is at most any convex combination of the weights of the paths. Specifically,

$$w(P_v) \leq \min(w(P_1), w(P_2), w(P_3), w(P_4)) \leq \frac{w(P_1)}{3} + \frac{w(P_2)}{3} + \frac{w(P_3)}{6} + \frac{w(P_4)}{6}.$$

We will now prove that

$$\frac{w(P_1)}{3} + \frac{w(P_2)}{3} + \frac{w(P_3)}{6} + \frac{w(P_4)}{6} \leq 1.5\ (\overline{vv_1} + \overline{vv_2} + \overline{vv_3}).$$

This simplifies to

$$\overline{v_1v_2} + \overline{v_2v_3} + \overline{v_3v_1} \leq 1.25\ \overline{vv_1} + 2(\overline{vv_2} + \overline{vv_3}),$$

which follows from Lemma 3.3.

Case 3. *v has 4 children, v_1, v_2, v_3, v_4, ordered clockwise around v.* Let v' be the point of intersection of the diagonals $\overline{v_1v_3}$ and $\overline{v_2v_4}$. Note that the diagonals do intersect because the polygon $v_1v_2v_3v_4$ is convex (follows from Corollary 3.2).

Let v_3 be the point that is furthest from v', among $\{v_1, v_2, v_3, v_4\}$. Consider the following two paths (see Fig. 4): $P_1 = [v, v_4, v_1, v_2, v_3], P_2 = [v, v_2, v_1, v_4, v_3]$.

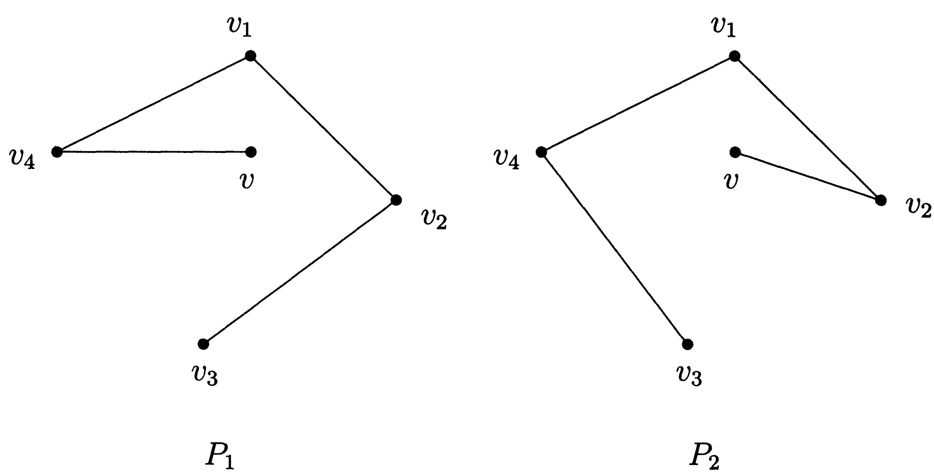

FIG. 4. *T_3, four children.*

Clearly,

$$w(P_v) \leq \min(w(P_1), w(P_2)) \leq \frac{w(P_1)}{2} + \frac{w(P_2)}{2}.$$

We will show that

$$\frac{1}{2}(w(P_1) + w(P_2)) \leq 1.5(\overline{vv_1} + \overline{vv_2} + \overline{vv_3} + \overline{vv_4}).$$

This simplifies to

$$\overline{v_1v_2v_3v_4} + (\overline{v_1v_2} + \overline{v_1v_4}) \leq 3(\overline{vv_1} + \overline{vv_3}) + 2(\overline{vv_2} + \overline{vv_4}). \tag{5}$$

We will first prove that

$$\overline{v_1v_2v_3v_4} + (\overline{v_1v_2} + \overline{v_1v_4}) \leq 3(\overline{v'v_1} + \overline{v'v_3}) + 2(\overline{v'v_2} + \overline{v'v_4}). \tag{6}$$

Once we prove (6), by the triangle inequality, we can conclude that (5) is true, since $\overline{vv_1} + \overline{vv_3} \geq \overline{v_1v_3} = \overline{v'v_1} + \overline{v'v_3}$ and $\overline{vv_2} + \overline{vv_4} \geq \overline{v_2v_4} = \overline{v'v_2} + \overline{v'v_4}$.

We prove (6) by contradiction. Suppose there exists a set of points which does not satisfy (6). Suppose we shrink $v'v_3$ by δ. The left side of the above inequality decreases by at most 2δ, whereas the right side of the inequality decreases by exactly 3δ. Therefore, as we shrink $v'v_3$, the inequality stays violated. Suppose $v'v_3$ shrinks and becomes equal to another edge $v'v_i$ for some $i \in \{1, 2, 4\}$. We now shrink both $v'v_3$ and $v'v_i$ simultaneously at the same rate. Again, it is easy to show that the inequality continues to be violated as $v'v_3$ and $v'v_i$ shrink. Hence we reach a configuration where three of the edges are equal.

Without loss of generality, the length of the three edges is 1 and the length of the fourth edge is some $\epsilon \leq 1$.

There are two subcases to consider. The first is when $v'v_1 = \epsilon$ and the second is when $v'v_2 = \epsilon$. (The case when $v'v_4 = \epsilon$ is the same as the second case.)

Case 3a. $v'v_1 = \epsilon$. We wish to prove that

$$\overline{v_1v_2v_3v_4} + (\overline{v_1v_2} + \overline{v_1v_4}) \leq 7 + 3\epsilon.$$

We want to show that the function $F(\epsilon) = \overline{v_1v_2v_3v_4} + (\overline{v_1v_2} + \overline{v_1v_4}) - 7 - 3\epsilon$ is nonpositive in the domain $0 \leq \epsilon \leq 1$. Simplifying, we get

$$F(\epsilon) = 2\overline{v_1v_2} + \overline{v_2v_3} + \overline{v_3v_4} + 2\overline{v_1v_4} - 7 - 3\epsilon.$$

Each of $\overline{v_iv_j}$ in the definition of F is a convex function of ϵ due to the following reason. Let p be the point closest to v_j on the line connecting v_i and v'. Observe that as v_i moves towards v', $\overline{v_iv_j}$ decreases if v_i is moving towards p and increases otherwise. Since F is a sum of convex functions minus a linear function, it is a convex function of ϵ. Therefore, it is maximized at either $\epsilon = 0$ or $\epsilon = 1$.

When $\epsilon = 1$, all four points are at the same distance from v'. If angle $\angle v_4v'v_1 = \alpha$ then F can be written as a function of a single variable α and it can be verified that F reaches a maximum value of $10\sqrt{0.8} - 10$, which is nonpositive.

When $\epsilon = 0$, $\overline{v_1v_2} = \overline{v_1v_4} = 1$. Simplifying we get $F = \overline{v_2v_3} + \overline{v_3v_4} - 3$, and it reaches a maximum value of $2\sqrt{2} - 3$, which is nonpositive (when $\epsilon = 0$, note that v_1 is the midpoint of the line segment v_2v_4).

Case 3b. $v'v_2 = \epsilon$. We wish to prove that

$$\overline{v_1v_2v_3v_4} + (\overline{v_1v_2} + \overline{v_1v_4}) \leq 8 + 2\epsilon.$$

We want to show that the function $F'(\epsilon) = \overline{v_1v_2v_3v_4} + (\overline{v_1v_2} + \overline{v_1v_4}) - 8 - 2\epsilon$ is nonpositive in the domain $0 \leq \epsilon \leq 1$.

As a function of ϵ, function F' is a sum of convex functions minus a linear function and thus is convex. Therefore, it is maximized at either $\epsilon = 0$ or $\epsilon = 1$.

The case $\epsilon = 1$ leads to the same configuration as in Case 3a.

When $\epsilon = 0$, $\overline{v_1v_2} = \overline{v_2v_3} = 1$. Here $F' = 2\overline{v_1v_4} + \overline{v_3v_4} - 5$. If angle $\angle v_4v'v_1 = \alpha$, then F' can be written as a function of a single variable α and it can be verified that F' reaches a maximum value of $5\sqrt{0.8} - 5$, which is nonpositive.

This concludes the proof of Lemma 3.5. □

The example in Fig. 5 shows that the 1.5 factor is tight for the algorithm in Fig. 2, modified according to the note following its description. The same example also shows that the 1.5 factor is tight for the unmodified algorithm since the unmodified algorithm never outputs a lighter tree than the modified algorithm. Each curved arc shown in Fig. 5 is actually a straight line and has been drawn curved for convenience. The vertex that is the child of the root has three children and is forced to drop one child. In doing so, the degree of its child goes to 4, and it in turn drops one of its children. The algorithm could make choices in such a way that the changes propagate through the tree and the tree T_3 output by the algorithm may be as shown in the figure. The ratio of the cost of the final solution to the cost of the MST can be made arbitrarily close to 1.5. See §5 for a discussion on the worst-case ratio between degree-3 trees and MSTs.

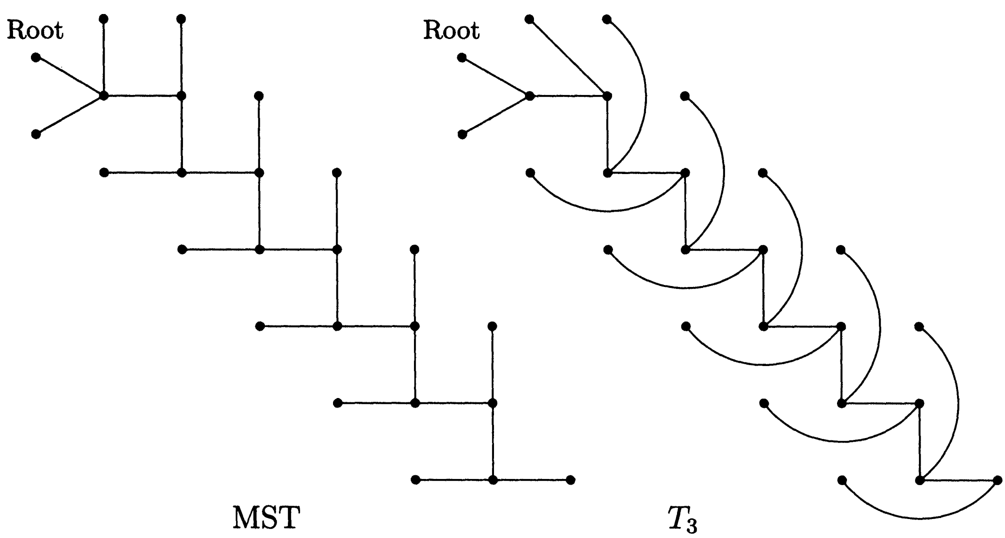

FIG. 5. *Bad example for algorithm in Fig. 2.*

3.3. Spanning trees of degree 4. We now assume that we are given a Euclidean minimum-spanning tree in which every vertex has degree at most 5. We show how to convert this tree to a tree in which every vertex has degree at most 4.

High level description. The basic idea is the same as in the previous algorithm. The difference is that we don't insist that each path P_v start at v. The tree is rooted at an arbitrary leaf. For each vertex v, the minimum-weight path P_v visiting v and all of v's children (not necessarily starting at v) is computed. The final tree T_4 consists of the union of the paths $\{P_v\}$. Again, for the analysis we think of each path P_v replacing the edges between v and its children in T.

TREE-4(V, T) — *Find a degree-4 tree of* V.
1 Root the MST T at a leaf vertex r.
2 For each vertex $v \in V$ do
3 Compute the shortest path P_v visiting v and all its children.
4 Return T_4, the tree formed by the union of the paths $\{P_v\}$.

FIG. 6. *Algorithm to find a degree-4 tree.*

LEMMA 3.7. *The algorithm in Fig. 6 returns a degree-4 spanning tree of the given set of points V.*

Proof. A proof by induction shows that T_4 is a tree. Each vertex v occurs in at most two paths and thus has degree at most 4. □

LEMMA 3.8. *Let v be a vertex in an MST T for a set of points in $\Re^2$. Let P_v be the shortest path visiting $\{v\} \cup child_T(v)$.*

$$w(P_v) \leq 1.25 \times \sum_{v_i \in child_T(v)} \overline{vv_i}.$$

From the above lemma, each path P_v weighs at most 1.25 times the net weight of the edges it replaces. Thus we have the following theorem.

THEOREM 3.9. *Let T be an MST of a set of points in $\Re^2$. Let T_4 be the spanning tree output by the algorithm in Fig. 6.*

$$w(T_4) \leq 1.25 \times w(T).$$

Proof of Lemma 3.8. The proof is similar to the proof of Lemma 3.5. As before, we consider cases depending on the number of children of v. The cases when v has no children, one child, or two children are trivial.

Case 1. *v has 3 children, v_1, v_2, v_3.* Let v_1 be the point that is closest to v, among its children. Consider the following four paths (see Fig. 7): $P_1 = [v_2, v_1, v, v_3], P_2 = [v_2, v, v_1, v_3], P_3 = [v_1, v, v_2, v_3]$, and $P_4 = [v_1, v, v_3, v_2]$.

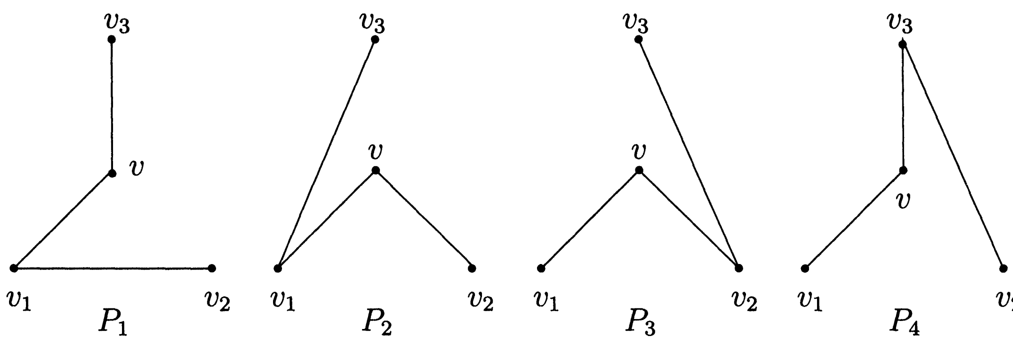

FIG. 7. *T_4, three children.*

Clearly,

$$w(P_v) \leq \frac{w(P_1)}{3} + \frac{w(P_2)}{3} + \frac{w(P_3)}{6} + \frac{w(P_4)}{6}.$$

We will show that

$$\frac{w(P_1)}{3} + \frac{w(P_2)}{3} + \frac{w(P_3)}{6} + \frac{w(P_4)}{6} \leq \frac{2+\sqrt{3}}{3}(\overline{vv_1} + \overline{vv_2} + \overline{vv_3}).$$

This proves the three-child case because $\frac{2+\sqrt{3}}{3}$ approximately equals 1.244 and is less than 1.25. This simplifies to

$$\frac{\overline{v_1v_2} + \overline{v_1v_3} + \overline{v_2v_3}}{3} + \overline{vv_1} + \frac{\overline{vv_2} + \overline{vv_3}}{2} \leq \frac{2+\sqrt{3}}{3}(\overline{vv_1} + \overline{vv_2} + \overline{vv_3}),$$

which further simplifies to

$$\overline{v_1v_2v_3} \leq (\sqrt{3} - 1)\overline{vv_1} + \left(\sqrt{3} + \frac{1}{2}\right)(\overline{vv_2} + \overline{vv_3}). \tag{7}$$

Since v_1 is the closest point to v, applying Lemma 3.3, we get

$$\overline{v_1v_2v_3} \leq (3\sqrt{3} - 4)\overline{vv_1} + 2(\overline{vv_2} + \overline{vv_3}),$$

and hence

$$\begin{aligned}\overline{v_1v_2v_3} &\le (\sqrt{3}-1)\overline{vv_1} + (2\sqrt{3}-3)\overline{vv_1} + 2(\overline{vv_2}+\overline{vv_3})\\ &\le (\sqrt{3}-1)\overline{vv_1} + (\sqrt{3}+\frac{1}{2})(\overline{vv_2}+\overline{vv_3}).\end{aligned}$$

This proves (7).

Case 2. *v has* 4 *children,* v_1, v_2, v_3, v_4. Assume that v_1 is the point that is closest to v among v's children. Let the order of the points be v_1, v_2, v_3, v_4 when we scan the plane clockwise from v starting from an arbitrary direction.

There are two cases, depending on whether v_4 or v_3 is the point that is furthest from v among its children. We first address the case when v_4 is the furthest point. (The proof for the case when v_2 is the point furthest from v is symmetric to the case when v_4 is the furthest point.)

Consider the following paths: $P_1 = [v_4, v_1, v, v_2, v_3]$ and $P_2 = [v_4, v_3, v, v_1, v_2]$ (see Fig. 8).

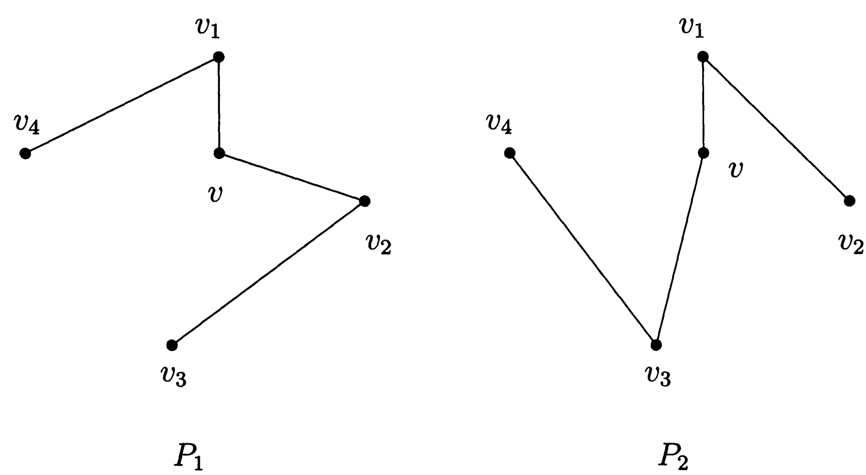

FIG. 8. *T_4, four children.*

The path P_v added by the algorithm is at most as heavy as the lighter of the paths P_1 and P_2. Hence

$$w(P_v) \le \min(P_1, P_2) \le \frac{w(P_1)+w(P_2)}{2}.$$

We will show that

$$\frac{w(P_1)+w(P_2)}{2} \le 1.25(\overline{vv_1}+\overline{vv_2}+\overline{vv_3}+\overline{vv_4}).$$

Simplifying, we need to show that

$$\frac{1}{2}(\overline{v_4v_1}+\overline{v_1v}+\overline{vv_2}+\overline{v_2v_3}+\overline{v_4v_3}+\overline{v_3v}+\overline{vv_1}+\overline{v_1v_2}) \le \frac{5}{4}(\overline{vv_1}+\overline{vv_2}+\overline{vv_3}+\overline{vv_4}).$$

Further simplifying, we get

$$\overline{v_1v_2v_3v_4} \le \frac{1}{2}\overline{vv_1} + \frac{5}{2}\overline{vv_4} + \frac{3}{2}(\overline{vv_2}+\overline{vv_3}).$$

Note that if it happens that v_3 was the farthest point from v among v's children, we get a similar equation with v_3 and v_4 being exchanged in the r.h.s of the equation. By symmetry, the case when v_2 is furthest is similar to v_4 being farthest.

Without loss of generality, $\overline{vv_3} \geq \overline{vv_2}$. The proof now proceeds in a manner similar to the proof of Lemma 3.3. If there is a configuration of points for which this equation is not true (the l.h.s exceeds the r.h.s) then we can move v_4, v_3 closer to v until $\overline{vv_2} = \overline{vv_3} = \overline{vv_4}$. In doing this, we decrease the l.h.s by at most $2(\overline{vv_4} - \overline{vv_2}) + 2(\overline{vv_3} - \overline{vv_2})$. Clearly, the r.h.s decreases by exactly $4(\overline{vv_4} - \overline{vv_2}) + 4(\overline{vv_3} - \overline{vv_2})$. This ensures that the l.h.s is still greater than the r.h.s. Hence without loss of generality, if there is a configuration for which our equation is not true, then there is a configuration with the property that $\overline{vv_4} = \overline{vv_3} = \overline{vv_2}$. We now show that when this property is true there is no counterexample.

By scaling, we may assume that $\overline{vv_4} = \overline{vv_3} = \overline{vv_2} = 1$ and $\overline{vv_1} = \epsilon$, where $\epsilon \leq 1$.

Note that (by Corollary 3.2) v was originally within the convex hull of its four children. Also (by Corollary 3.2), every child is on the convex hull. These properties are both maintained by the above shrinking steps.

We now wish to prove that

$$\overline{v_1v_2v_3v_4} \leq \frac{11}{2} + \frac{1}{2}\epsilon.$$

It is easily shown using elementary calculus that for any ϵ such that v_1 is on the convex hull of the points $\{v_1, \ldots, v_4\}$, rotating v_1 and v_3 around v until $\angle v_1vv_2 = \angle v_1vv_4$ (see Fig. 9) and $\angle v_2vv_3 = \angle v_4vv_3$ does not decrease the perimeter. Also, it maintains that v_1 is on the convex hull. Assume the two pairs of angles are equal, and define $F(\epsilon) = \overline{v_1v_2v_3v_4} - \epsilon/2 - 11/2$. We will show that F is nonpositive over the domain of possible ϵ's.

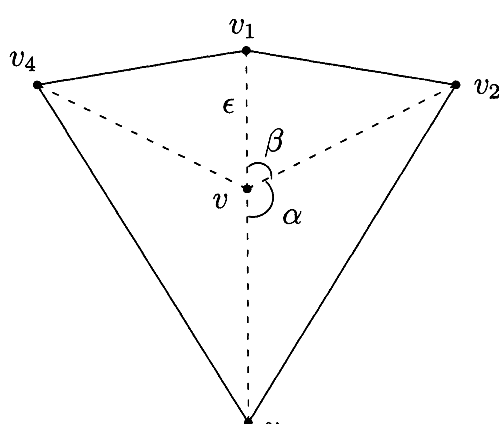

FIG. 9. *Figure to illustrate degree-4 case.*

As a function of ϵ, function F is a sum of convex functions minus a linear function and thus is convex. Therefore, F is maximized either when $\overline{vv_1} = 1$ or when v_1 is the midpoint of edge $\overline{v_2v_4}$ (since v_1 is on the convex hull, v_1 cannot cross the edge; hence this interval contains all possible values for ϵ).

In the first case, all four points lie on a unit circle with center at v. For any four such points, it is easily proven using calculus that $\overline{v_1v_2v_3v_4}$ is maximized when the four points are the vertices of a square at $4\sqrt{2} \approx 5.66$. Thus, $F(1) < 0$.

In the second case, $\overline{v_1v_2v_3v_4} = \overline{v_2v_3v_4}$. As noted previously, this is at most $3\sqrt{3} \approx 5.2$. Thus, $F(\epsilon) < 0$.

We now deal with the case when v_3 is the furthest point. In this case, we take the paths $P_1 = [v_4, v_1, v, v_2, v_3]$ and $P_2 = [v_3, v_4, v, v_1, v_2]$. The path P added by the algorithm is at most as heavy as the lighter of the paths P_1 and P_2. Hence,

$$w(P) \leq \min(P_1, P_2) \leq \frac{w(P_1) + w(P_2)}{2}.$$

Simplifying, we get

$$\overline{v_1v_2v_3v_4} \leq \frac{1}{2}\overline{vv_1} + \frac{5}{2}\overline{vv_3} + \frac{3}{2}(\overline{vv_2} + \overline{vv_4}).$$

The proof of this is identical to the proof of the previous case. □

4. Points in higher dimensions. We show how to compute a degree-3 tree (T_3) when the points are in arbitrary dimension $d \geq 3$. The algorithm for computing the tree is similar to the algorithm for computing degree-3 trees in the plane—the tree T_3 is formed by rooting the MST and taking the union of the paths $\{P_v\}$, where each P_v is the shortest path starting at v and visiting all of the children of v in the rooted MST. It is known that any Euclidean MST has constant degree [17] (for any fixed dimension), so that the algorithm still requires only linear time. The bound on the weight of T_3 is similar, except that v may have more children. We prove that regardless of the number of children that v has, the weight of P_v is at most 5/3 the weight of the edges that it replaces:

LEMMA 4.1. *Let $\{v, v_1, v_2, \ldots, v_k\}$ be a set of arbitrary points in $\Re^d$. There is a path P, starting at v, that visits all the points $v_1, v_2, \ldots, v_k$ such that*

$$w(P) \leq \frac{5}{3}\sum_{i=1}^{k}\overline{vv_i}.$$

Proof. We prove this by induction on the degree of v. Sort the points in increasing distance from v as $v_1, \ldots, v_k$. Let $v = v_0$. The lemma is trivially true when $k = 0, 1, 2$. Let us assume that the lemma is true for all values of k up to some $\ell \geq 2$. Consider $k = \ell + 1$. By the induction hypothesis, the claim is true when v has $k - 3$ children; hence we can find a path P' that starts at v and visits all vertices v_i ($i = 1, \ldots, k-3$) (not necessarily in that order) such that $w(P') \leq (5/3)\sum_{i=1}^{k-3}\overline{vv_i}$. Let v_j be the last vertex on the path P'. We add the cheapest path P'' that starts at v_j and visits v_{k-2}, v_{k-1}, and v_k (again, not necessarily in that order). This path together with P' will form a path that starts at v and visits all vertices adjacent to v. We now show that

$$w(P'') \leq \frac{5}{3}(\overline{vv_{k-2}} + \overline{vv_{k-1}} + \overline{vv_k}). \tag{8}$$

This suffices to prove the lemma. Let $P_1, \ldots, P_6$ be the six possibilities for P''. Clearly,

$$w(P'') \leq \frac{1}{6}\sum_{i=1}^{6} w(P_i).$$

We will prove that

$$\frac{1}{6}\sum_{i=1}^{6} w(P_i) \leq \frac{5}{3}(\overline{vv_{k-2}} + \overline{vv_{k-1}} + \overline{vv_k}).$$

This simplifies to

$$2\ \overline{v_{k-2}v_{k-1}v_k} + \sum_{i=k-2}^{k} \overline{v_j v_i} \leq 5(\overline{vv_{k-2}} + \overline{vv_{k-1}} + \overline{vv_k}). \tag{9}$$

Notice that if the above equation is not true, we can "shrink" all the v_i ($i = k-2, k-1, k$) until $\overline{vv_j} = \overline{vv_{k-2}} = \overline{vv_{k-1}} = \overline{vv_k}$. Assume that $\delta = (\overline{vv_{k-2}} - \overline{vv_j}) + (\overline{vv_{k-1}} - \overline{vv_j}) + (\overline{vv_k} - \overline{vv_j})$. This can be done because the r.h.s decreases by 5δ and the l.h.s decreases by at most 5δ. If the above equation is not true, then it is also not true when the distance from v to all the points is the same. By scaling, we can assume that the distance of the points from v is 1. We call this a canonical configuration. The following proposition is implied by Lillington's work [13] and helps in completing the proof.

PROPOSITION 4.2. *Let A, B, C, and D be points on a unit sphere in d dimensions, $d \geq 3$. The function $F = \overline{AB} + \overline{AC} + \overline{AD} + \overline{BC} + \overline{CD} + \overline{BD}$ reaches a maximum value of $4\sqrt{6}$ when the points A, B, C, and D form a regular tetrahedron.*

We will now show that (9) is satisfied by the canonical configuration. The left side of (9) can be written as the sum of the sides of the tetrahedron formed by the points $\{v_k, v_{k-1}, v_{k-2}, v_j\}$ and the sum of the sides of the triangle formed by the points $\{v_k, v_{k-1}, v_{k-2}\}$. These points lie on a sphere whose center is v. By Lemma 4.2, the first sum is bounded by $4\sqrt{6}$. The second sum is bounded by $3\sqrt{3}$. Hence the left side of (9) is bounded by $4\sqrt{6} + 3\sqrt{3}$, which is about 14.994. The right side of (9) is 15. Hence (9) is satisfied by the canonical configuration and therefore all configurations. This concludes the proof of Lemma 4.1. □

Remark. The algorithm outlined earlier runs in linear time only when d, the number of dimensions, is a constant. The algorithm can be modified to run in linear time for all d as follows. Observe that in the proof of Lemma 4.1, we considered the neighbors of v only three at a time. Therefore, the algorithm could also group vertices into sets of three each, based on the distance from v, and inductively construct the path as in the proof of the lemma. This algorithm would have the same performance guarantee (5/3) as the earlier algorithm for constructing a degree-3 tree and in addition have the added advantage of running in linear time for all dimensions.

5. Conclusions. We have given a simple algorithm for computing a degree-3 (degree-4) tree for points in the plane that is within 1.5 (1.25) of an MST of the points. An extension of the algorithm finds a degree-3 tree of an arbitrary set of points in d dimensions within 5/3 of an MST. If an MST of the points is given as part of the input, our algorithms run in linear time. All our proofs are based on elementary geometric techniques.

Though our algorithms improve greatly the best-known ratios for each of the respective problems, there are still large gaps between the ratios that we obtain and the best bounds that we think are achievable. For example, in the case of points in the plane, consider the ratio of the weight of a minimum weight degree-3 tree to the weight of an MST. The worst example that we can obtain for this ratio is $\frac{\sqrt{2}+3}{4} \approx 1.104$ (with five points, where four of the points are at the corners of a square and the fifth point is in the middle). There is a large gap between this and the ratio of 1.5 obtained by our algorithm. Is 1.104 the worst-case ratio? Are there polynomial time algorithms which obtain factors better than 1.5? Notice that the performance ratio obtained by our algorithm on the example in Fig. 5 is highly sensitive to the vertex chosen as the root. One potential algorithm is to simply try all possible vertices as the root, and to pick the tree of minimum weight. Does such an algorithm have a better performance guarantee?

For the problem of finding degree-4 trees, our algorithm obtains a ratio of 1.25. Unlike degree-3 trees, we are unable to show that this ratio is tight for the algorithm. Can the factor of 1.25 for the algorithm be improved? The worst example for the ratio between a minimum-weight degree-4 tree and an MST that we can obtain is about 1.035 (five points on the vertices of a regular pentagon with a sixth point in their centroid). Are there examples with worse ratios?

Problems of approximating degree-k trees in higher dimensions and in general metric spaces within factors better than 2 are still open.

Acknowledgments. We thank Andras Bezdek for telling us about [13]. We thank Karoly Bezdek and Bob Connelly for useful discussions and the committee members of the 1994 Syposium on the Theory of Computing for simplifying the proof of Lemma 3.3 and for pointing out [14].

REFERENCES

[1] J. L. Bentley, Communicated by David Johnson.

[2] D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. P. Williamson, *A note on the prize collecting traveling salesman problem*, Math. Programming, 59 (1993), pp. 413–420.

[3] N. Christofides, *Worst-case analysis of a new heuristic for the traveling salesman problem*, Tech. report 388, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, PA, 1975.

[4] G. Das and P. J. Heffernan, *Constructing degree-3 spanners with other sparseness properties*, in Proc. 4th Annual International Symposium on Algorithms and Computation, Lecture Notes in Comput. Sci. 762, Springer-Verlag, Berlin, New York, 1993, pp. 11–20.

[5] D.-Z. Du and F. K. Hwang, *A proof of the Gilbert–Pollak conjecture on the Steiner ratio*, Algorithmica, 7 (1992), pp. 121–136.

[6] D.-Z. Du, Y. Zhang, and Q. Feng, *On better heuristic for Euclidean Steiner minimum trees*, in Proc. 32nd Annual Symposium on Foundations of Computer Science, IEEE Press, Piscataway, NJ, 1991, pp. 431–439.

[7] T. Fischer, *Optimizing the degree of minimum weight spanning trees*, Tech. report 93-1338, Department of Computer Science, Cornell University, April 1993.

[8] G. N. Frederickson and J. JáJá, *On the relationship between the biconnectivity augmentation and traveling salesman problems*, Theoret. Comput. Sci., 19 (1982), pp. 189–201.

[9] M. Fürer and B. Raghavachari, *Approximating the minimum-degree Steiner tree to within one of optimal*, J. Algorithms, 17 (1994), pp. 409–423.

[10] M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W. H. Freeman, San Francisco, 1979.

[11] M. X. Goemans and D. P. Williamson, *A general approximation technique for constrained forest problems*, in Proc. 3rd Annual ACM–SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, 1992, pp. 307–316; SIAM J. Comput., 24 (1995), pp. 296–317.

[12] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter, *Hamilton paths in grid graphs*, SIAM J. Comput., 11 (1982), pp. 676–686.

[13] J. N. Lillington, *Some extremal properties of convex sets*, Math. Proc. Cambridge Philos. Soc., 77 (1975), pp. 515–524.

[14] C. Monma and S. Suri, *Transitions in geometric minimum spanning trees*, Discrete Comput. Geom., 8 (1992), pp. 265–293.

[15] C. H. Papadimitriou and U. V. Vazirani, *On two geometric problems related to the traveling salesman problem*, J. Algorithms, 5 (1984), pp. 231–246.

[16] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt III, *Many birds with one stone: Multi-objective approximation algorithms*, in Proc. 25th Annual ACM Symposium on the Theory of Computing, Association for Computing Machinery, New York, 1993, pp. 438–447.

[17] G. Robins and J. S. Salowe, *On the maximum degree of minimum spanning trees*, in Proc. 10th Annual ACM Symposium on Computational Geometry, Association for Computing Machinery, New York, 1994, pp. 250–258; Discrete Comput. Geom., 14 (1995), pp. 151–166.

[18] J. S. Salowe, *Euclidean spanner graphs with degree four*, in Proc. 8th Annual ACM Symposium on Computational Geometry, Association for Computing Machinery, New York, 1992, pp. 186–191; Discrete Appl. Math., 54 (1994), pp. 55–66.