An O(log k) approximation algorithm for the k minimum spanning tree problem in the plane

Naveen Garg* Dorit S. Hochbaum†

Abstract

Given n points in the Euclidean plane, we consider the problem of finding the minimum tree spanning any k points. The problem is **NP**-hard and we give an $O(\log k)$-approximation algorithm.

1 Introduction

A cable company is permitted to service k cities in a state with n cities. To minimise the cost of installation the company wishes to find a set of cities that are "close". Since we need to connect these cities, a natural measure of "closeness" is the weight of the minimum tree spanning them where the weight of the edge (i, j) is the euclidean distance, $d(i, j)$, between cities i and j. More formally, the problem that we address in this paper is the following.

> **Problem k-EMST** Given n points in the Euclidean plane, find the minimum weight tree spanning at least k points.

The motivation for the problem arose in bidding for licenses to operate oil rigs within contiguous regions. That application and a facility layout application are described by Fischetti, Hamacher, Jornsten and Maffioli in [5]. They also prove that the problem is **NP**-hard. Ravi, Sundaram, Marathe, Rosenkrantz and Ravi [12] showed that it is **NP**-hard to compute the minimum weight tree spanning k points in the *plane*. Note that the problem is polynomially solvable for fixed k; for every set of k points we compute the minimum tree spanning the set and pick the set for which the weight of this tree is minimum. The problem is polynomial when the underlying graph is a tree [9]. In [6] there is a further study of the problem on general graphs, and some integer programming formulations and solution techniques are demonstrated.

Ravi et.al. [12] were the first to establish an approximation worst case bound for the problem. Their paper describes an algorithm that achieves an approximation factor of $O(k^{1/4})$. It also gives a polynomial time exact solution for the special case when the points are on the boundary of a convex region in the plane.

*Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany
†Industrial Engineering and Operations Research, University of California, Berkeley, CA 94720

The main result of this paper is an $O(\log k)$ approximation to the k-EMST problem. We use a family of rectangular grids to identify the set of k points. The set of points we pick is such that they occupy only a small number of cells in each grid. We show that for a particular choice of grids this implies that the minimum tree spanning the point set is small. Subsequent to the first appearance of this paper, there has been significant progress in improving the approximation guarantee for this problem. The approximation factor was improved to $O(\frac{\log k}{\log \log n})$ by Eppstein [4], to $O(1)$ by Blum, Chalasani and Vempala [3] and more recently to $2\sqrt{2}$ by Mitchell [10].

2 Related problems

The problem of computing the minimum tree spanning any k points can be viewed in the general setting of finding a "best" subgraph spanning k vertices and satisfying certain properties. Clearly, all problems for which the properties can be checked in polynomial time are polynomially solvable for fixed k, by examining all k point subsets.

In particular if we require that the subgraph be connected and of the least possible weight, then we are really looking for the minimum weight tree that spans k vertices; an extension of the k-EMST problem to general graphs. A $3\sqrt{k}$-approximation algorithm for this problem was first given by [12] and this has been improved recently to a $O(\log^2 k)$-approximation algorithm by Awerbuch, Azar, Blum and Vempala [2].

If the subgraph we are looking for is the one with the maximum weight then the problem is **NP**-hard as the k-clique problem can be reduced to it [13]. The best known approximation ratio for this *heaviest subgraph* problem is $\tilde{O}(n^{0.3885})$ and is due to Kortsarz and Peleg [8]. For trees the problem can be solved in time $O(nk^2)$ using dynamic programming [7]. Similar to the heaviest subgraph problem is the *lightest induced subgraph* problem where we wish to pick k vertices such that the subgraph induced is of the minimum weight. Once again this problem is **NP**-hard (the k-independent set problem can be reduced to it); the geometric version of this problem has been studied in [1] and the graph version in [11].

3 Lower bounds

An important issue in the design of approximation algorithms for optimization problems is that of the bounding technique used. For the purpose of analysing the algorithm we need to compare the cost of its solution to the optimum. But since, computing the value of the optimum is itself **NP**-hard we need a lower bound on the optimum (assuming a minimisation problem) against which we can compare our solution. Further, this lower bound should be close to the optimum; if for instance, optimum value $\geq \alpha \cdot$(lower bound), then any approximation algorithm that uses this lower bounding technique can at best guarantee an approximation factor of α. Thus, we need to look for good lower bounding techniques that mimic the optimum.

We first discuss two lower bounding techniques due to [12]. Our technique can be viewed as a refinement of these lower bounds.

3.1 Diameter

Let V be the set of n points and $V^* \subseteq V$ be the k points picked by the optimum. We denote by $d(i,j)$ the Euclidean distance between points i and j. The diameter of a set, $S \subseteq V$, is the maximum distance between any two points of S and is denoted by $diam(S)$.

For every pair of points i, j construct a circle C_{ij} with center as the midpoint of line segment (i,j) and diameter $\sigma = \sqrt{3}d(i,j)$. This choice of center and diameter ensures that the circle corresponding to the farthest points[1] of V^* contains all of V^*. Let C_{pq} be the circle with the minimum diameter that contains at least k points. Clearly, $d(p,q) \leq diam(V^*) \leq$ weight of minimum tree spanning V^*. Thus, $d(p,q)$ is a lower bound on the weight of the minimum tree spanning k points. [12] show how to obtain an $O(\sqrt{k})$ approximation using this lower bounding technique.

A disadvantage of this lower bounding technique is that the lower bound is always less than the diameter of the optimum point set, V^*. Thus if V^* were such that the minimum tree spanning V^* was large compared to $diam(V^*)$ then we would have a large gap between the optimum value and the lower bound. This, for example, would be the case when V^* is a set of evenly spaced points in a square. The next lower bounding technique, also due to [12], takes care of this shortcoming.

3.2 The Grid

Lemma 3.1 ([12]) *Consider a rectangular grid, G, each cell of which is a square of side x. If t is the minimum number of cells containing k points then the minimum weight tree spanning any k points has weight $\Omega(tx)$.*

Ravi et.al. [12] combine this lower bound with diameter considerations and obtain an $O(k^{1/4})$ approximation to the optimum.

Note that if the optimum point set V^* is such that all points of V^* are contained in a small number of cells of G but these cells are far apart, then the value of this lower bound could be much less than the optimum. We obtain a better lower bound by considering grids of different cell sizes.

4 The Approximation Algorithm

The algorithm works by "guessing" for each pair of points, that they are the furthest away in the optimal solution set. For every pair of points i, j construct a circle C_{ij} with center as the midpoint of the line segment (i,j) and diameter $\sigma = \sqrt{3}d(i,j)$. This choice of center and diameter ensures that any point set for which (i,j) are the farthest points lies completely within C_{ij}. Let Q_{ij} be a square of side σ circumscribing C_{ij}. If p, q are the farthest points in the optimal point set V^*, then all points of V^* are contained in Q_{pq}. Thus, it suffices to solve the problem independently in each square Q_{ij} (by restricting ourselves to the points contained in the square) and then picking the minimum tree. From now on we limit our discussion to the square Q_{pq} of side $\sigma = \sqrt{3}d(p,q)$. For the sake of brevity we denote this square by Q and let V denote the set of points contained in it.

[1] If p, q are the two farthest points in V^* and v is another point in V^* then $d(v,p) \leq d(p,q)$. Also, $d(v,q) \leq d(p,q)$ and therefore the distance of v from the midpoint of the line segment joining p and q is at most $\frac{\sqrt{3}}{2}d(p,q)$

We need to find a set of k points in Q that are "close", in the sense that the weight of the minimum tree spanning the points is small. To this end we define a potential function on the point set, $P : 2^V \rightarrow \mathcal{R}^+$. Our choice of potential is such that a set of points is "close" if its potential is small.

Let G_i be a rectangular grid in Q, each cell of which is a square of side x_i. The *size* of grid G_i is the size of its cell i.e x_i.

Definition 4.1 *The G_i-*potential of a set $S \subseteq V$, *denoted by $G_i(S)$, is equal to $x_i t_i$ where t_i is the number of cells of G_i that contain points of S.*

As remarked earlier this definition of potential is not sufficient as we could have a set of k points that are contained in only a few cells of G_i which, however, are far apart. This set of points has a small potential with respect to this grid even though the minimum tree spanning this set is large. To take care of this we define the potential of a set with respect to a collection of grids.

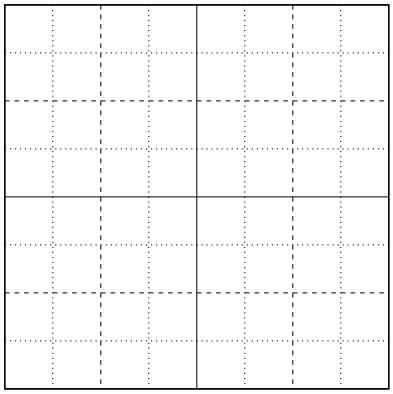

Figure 1: Grids G_0, G_1, G_2

Note that any set of points has a potential σ with respect to the square Q. Further, the G_i-potential of any set of k points is less than σ if grid G_i has a size less than $\frac{\sigma}{k}$. Hence, we get no advantage by considering a grid of size less than $\frac{\sigma}{k}$ in our family of grids. Let G_0 be a rectangular grid of size $x_0 = \frac{\sigma}{k}$ in Q. Define G_i to be the grid, each cell of which is a square composed of four cells of grid G_{i-1}. If x_i denotes the size of G_i then $x_i = 2x_{i-1} = 2^i x_0$. Since $x_{\log k} = 2^{\log k} x_0 = k x_0 = \sigma$, $G_{\log k}$ is the square Q[2]. Thus we have $\log k$ different grids $G_0, G_1 \ldots G_{\log k - 1}$. Figure 1 shows grids G_0, G_1 and G_2 drawn with dotted, dashed and solid lines respectively.

Definition 4.2 *The* potential of a set S *is the sum of the G_i-potentials of S i.e.*

$$P(S) = \sum_{i=0}^{\log k - 1} G_i(S)$$

Our algorithm essentially finds a set of k points with the minimum potential and computes the minimum tree spanning the points.

[2]Throughout this paper we assume that k is a power of 2; for general k all arguments apply with $\log k$ replaced by $\lceil \log k \rceil$.

Algorithm E-MST(k);
for all pairs of points i, j **do**

1. Construct a circle centered at the midpoint of the line segment (i, j) with diameter $\sigma = \sqrt{3}d(i, j)$.

2. **if** circle contains at least k points **then**

{*Let Q be a square of side σ circumscribing the circle.* }

2.1. $S^* = \text{minpot}(Q)$

{*find set with minimum potential in Q* }

2.2. Compute the minimum tree spanning S^*

Output the minimum weight tree found in Step 2.2 and the set S^* corresponding to it.
end.

In Section 6 we show a polynomial time algorithm for the procedure minpot(Q) which, given the square Q finds a set of k points in Q that has the minimum potential. In the next section we prove, formally, our claim that the potential of a point set is small iff the weight of the minimum tree spanning this set is small and show how this leads to the $O(\log k)$ approximation factor.

5 The approximation guarantee

We first show that a "close" set of points has a small potential.

Lemma 5.1 *If V^* is the optimum point set then*

$$P(V^*) = \sum_{i=0}^{\log k - 1} G_i(V^*) \leq 8 \log k \cdot OPT$$

where OPT is the weight of the minimum tree spanning V^*.

Proof: Consider a grid G_i of size x_i and let t_i cells of this grid contain all the points of V^*. We first prove that the G_i-potential of V^*, $G_i(V^*)$, is at most $8 \cdot \text{OPT}$.

We have two cases depending on the value of t_i.

1. $t_i \leq 4$
   Since the coarsest grid has a size of $\frac{\sigma}{2}$, $x_i \leq \frac{\sigma}{2}$. Also, $\sigma = \sqrt{3}d(p, q) \leq \sqrt{3} \cdot \text{OPT}$ and therefore
   $$G_i(V^*) = x_i t_i \leq 2\sigma \leq 2\sqrt{3} \cdot \text{OPT} < 8 \cdot \text{OPT}$$

2. $t_i > 4$
   We color the cells of the grid as follows: assign every cell a color depending on whether the row/column that the cell belongs to is odd/even. Since these are four different possibilities we use only 4 different colors and no two cells that are adjacent or diagonally adjacent have the same color. Since the points of V^* lie in t_i cells, there are at least $\frac{t_i}{4}$ cells that contain points from V^* and belong to the same color class. To connect two cells from the same color

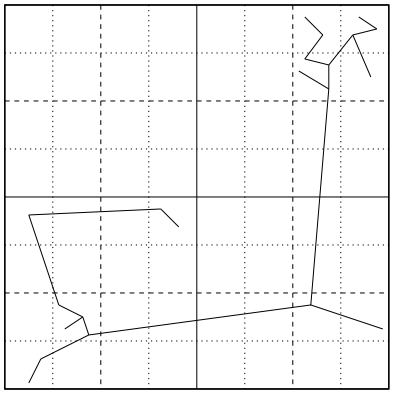

Figure 2: Building the spanning tree

class requires an edge of length at least x_i. Hence the tree spanning V^* must have length at least $x_i(\frac{t_i}{4} - 1)$ which, since $t_i > 4$, is at least $\frac{x_i t_i}{8}$. Therefore

$$G_i(V^*) = x_i t_i \leq 8 \cdot \mathrm{OPT}$$

Since $G_i(V^*) \leq 8 \cdot \mathrm{OPT}$ for every grid G_i, $0 \leq i \leq \log k - 1$, the Lemma follows. ■

The following Lemma proves the converse i.e. that a set of points with a small potential is "close".

Lemma 5.2 *The minimum tree spanning any set, S, of k points has weight at most $\sqrt{2}(\sum_{i=0}^{\log k - 1} G_i(S))$.*

Proof: We show this by constructing a tree spanning S of weight no more than the claimed bound. The tree is built in a hierarchical manner; we first connect all points of S contained in a cell of grid G_0 and then add edges across cells of G_0 to connect points of S in a cell of G_1 and so on (Figure 2).

We begin with the grid G_0 and for each cell of this grid pick a tree over the points of S contained in the cell. Consider a cell containing m points of S. Since no edge picked has length more than the diameter of the cell ($\sqrt{2}x_0$) the total weight of edges picked from this cell is $\sqrt{2}x_0(m-1)$. Hence the total weight of edges picked from all cells is at most $\sqrt{2}x_0(k - t_0)$ where t_0 is the number of cells of G_0 containing points from S. But $x_0 t_0 = G_0(S)$ and therefore the weight of edges picked in this step is $\sqrt{2}(x_0 k - G_0(S)) = \sqrt{2}(\sigma - G_0(S))$.

Our algorithm for connecting points of S proceeds in iterations. At the start of the i^{th} iteration all points of S contained in a cell of G_{i-1} are connected. In this iteration we add edges across cells of G_{i-1} so as to ensure that for each cell of G_i, all points of S contained in the cell are connected.

Claim 5.1 *The total weight of edges picked in the i^{th} iteration is at most $\sqrt{2}(2G_{i-1}(S) - G_i(S))$.*

Proof: Consider a cell of G_i. Of the four cells of G_{i-1} contained in this cell of G_i, let m ($0 \leq m \leq 4$), cells contain points from S. Thus we need to add $m - 1$ edges; each edge is of weight at most the diameter of the cell ($\sqrt{2}x_i$). Thus the weight of edges picked from this cell is $\sqrt{2}x_i(m-1)$. Hence, the total weight of edges picked from all cells of G_i is $\sqrt{2}x_i(t_{i-1} - t_i)$, where t_{i-1}, t_i are the number

of cells in G_{i-1}, G_i respectively that contain points from S. Since $x_i = 2x_{i-1}$, the total weight picked in this step is at most

$$\begin{aligned}\sqrt{2}x_i(t_{i-1} - t_i) &= \sqrt{2}(2x_{i-1}t_{i-1} - x_i t_i) \\ &= \sqrt{2}(2G_{i-1}(S) - G_i(S))\end{aligned}$$

■

Thus by the end of $\log k$ iterations we have a tree spanning all points of S. The total weight of this tree is at most $\sqrt{2}(\sigma - G_0(S) + 2G_0(S) - G_1(S) + \ldots + 2G_{\log k - 1}(S) - G_{\log k}(S))$, which is equal to $\sqrt{2}\sum_{i=0}^{\log k - 1} G_i(S)$. This completes the proof of Lemma 5.2. ■

Having proven that a set of points has a small potential iff it has a small tree spanning it (Lemmas 5.1 and 5.2) we now show that the minimum tree spanning S^* has weight at most $O(\log k)$ OPT. Recall that S^* is a set of k points from Q with the minimum potential. Since the optimal point set, V^*, is also contained in Q, $P(S^*) \leq P(V^*)$ and therefore

$$\begin{aligned}\text{weight of minimum tree spanning } S^* &\leq \sqrt{2}P(S^*) \\ &\leq \sqrt{2}P(V^*) \\ &\leq 8\sqrt{2}\log k \cdot \text{OPT} \\ &= O(\log k) \cdot \text{OPT}\end{aligned}$$

The only thing that now remains is to find the set S^* in polynomial time.

6 Finding the set with minimum potential

In this section we show a polynomial time algorithm for finding a "close" set of k points with the minimum potential. Our algorithm uses Dynamic Programming to identify this set.

Consider a cell in grid G_i and associate with this cell a list, L, of k elements. The p^{th} element in this list is the minimum total potential with respect to grids $G_0, G_1 \ldots G_i$ of any set of p points contained in this cell. Formally,

$$L(p) = \min_{S:|S|=p} \sum_{j=0}^{i} G_j(S)$$

where S is contained in the cell of G_i being considered. Note that the 0^{th} element of any list is always zero.

The lists for cells of G_0 can be constructed immediately. If a cell of G_0 contains m points then the elements $L(1), L(2) \ldots L(m)$ are set to x_0 while the remaining elements are assigned a suitably large value, say ∞. Given lists for cells of G_{i-1} we can obtain the list for a cell in grid G_i in the following manner. Consider the four cells of G_{i-1} that make up this cell of G_i. To obtain the p^{th} element we need to find the best way of splitting p ($p = p_1 + p_2 + p_3 + p_4$) such that picking p_1, p_2, p_3, p_4 points respectively from these four cells minimises the total potential with respect to grids $G_0, G_1 \ldots G_{i-1}$. This minimisation can be done in polynomial time by considering all 4-partitions of the number p and using the lists for cells of G_{i-1} to find the total potential with respect to grids $G_0, G_1 \ldots G_{i-1}$

for each partition. The p^{th} element for the cell of G_i is then the sum of this value and x_i, the potential with respect to grid G_i. Thus, for $p \geq 1$,

$$L(p) \quad = \quad \min_{p_1,p_2,p_3,p_4} \{L_1(p_1) + L_2(p_2) + L_3(p_3) + L_4(p_4)\} + x_i$$

where $p_1 + p_2 + p_3 + p_4 = p$ and L_1, L_2, L_3, L_4 are lists for the four cells of G_{i-1} contained in this cell.

The list L, can be computed more efficiently in $O(k^2)$ time as follows.

$$\begin{aligned} L(p) &= L_1(p) \\ L(p) &= \min_{0 \leq i \leq k} L(i) + L_2(p-i) \\ L(p) &= \min_{0 \leq i \leq k} L(i) + L_3(p-i) \\ L(p) &= \min_{0 \leq i \leq k} L(i) + L_4(p-i) \end{aligned}$$

and for $p \geq 1$

$$L(p) = L(p) + x_i$$

procedure minpot(Q);
    for all cells of G_0 **do**
        for $p := 1$ to k **do**
            if cell contains more than p points **then**
                $L(p) = x_0$
            else
                $L(p) = \infty$
    for $i := 1$ to $\log k$ **do**
        for all cells of G_i **do**
            Compute list L.
end.

Lemma 6.1 *The k^{th} element in the list for Q corresponds to a set of points with the minimum potential.*

Proof: Let S^* be the set of points corresponding to the kth element in the list for Q. By the manner in which we have defined the lists it follows that S^* has the minimum total potential with respect to grids $G_0, G_1 \ldots G_{\log k}$ from amongst any set of k points contained in Q, i.e.

$$\sum_{i=0}^{\log k} G_i(S^*) \leq \sum_{i=0}^{\log k} G_i(S)$$

where $|S^*| = |S| = k$ and $S \subseteq V$. Since, $G_{\log k}(S) = \sigma$ for any set $\phi \neq S \subseteq V$, S^* also minimises $P(\cdot) = \sum_{i=0}^{\log k - 1} G_i(\cdot)$ and therefore is a set with minimum potential. ■

Since the total number of cells in all the grids put together is $1^2 + 2^2 + 4^2 + \ldots + k^2 = \mathrm{O}(k^2)$, and computation of each list takes $O(k^2)$ time, the procedure minpot() has a running time complexity of $\mathrm{O}(k^4)$.

7 A Tight Example

The example given here illustrates that the $\log k$ gap between the lower and upper bounds is in fact attainable for a family of problems.

In this paper we use the diameter and the grid lower bounds to arrive at an $O(\log k)$-approximation to the k-EMST problem. This differs from [12] in that instead of considering just one grid, as in [12], we consider a collection of $\log k$ grids and look at the average lower bound with respect to these grids for every collection of k points. An alternative strategy would be to look at the maximum lower bound provided by any one of these $\log k$ grids since this would be at least as large as the average. The following example (Figure 3) is a collection of k points in a square of side σ such that each of the $\log k$ grids provides exactly the same lower bound, i.e. σ, and the weight of the minimum tree spanning these points is $\sigma \log k$. This shows that considering the maximum lower bound provided by any grid is no better than considering the average.

The points are arranged so that each cell of G_i which contains any points has them in exactly two cells of G_{i-1}. This implies that $t_i = \frac{k}{2^i}$ and so the potential of this set of points with respect to each grid is exactly σ.

Let the four cells of G_{i-1} in a cell of G_i be numbered 1,2,3,4 in a clockwise order starting at the top-left cell. Further, we distribute the points so that the two cells of G_{i-1} that the points of G_i are contained in are the cells numbered 2 and 4 if i is even and 1 and 3 if i is odd. This gives an arrangement of points as shown in Figure 3 and it is easy to see that the minimum steiner tree spanning these points is of length $O(\sigma \log k)$.

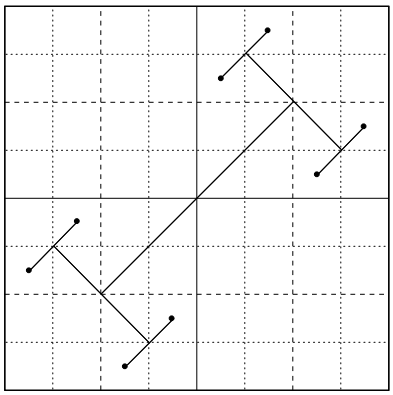

Figure 3: A tight example

8 On a variant of the travelling salesman problem

We now consider an interesting variant on the travelling salesman problem. Consider a travelling salesman who wishes to visit the maximum number of cities on a given amount of fuel. The bound on the fuel is equivalent to placing a bound on the total distance travelled by the salesman. Once again we assume that the cities are points in a plane and the distance between two cities is the euclidean distance between the corresponding points. More formally,

Problem Given n points in the Euclidean plane and a bound B, find a cycle of length at most B that includes the maximum number of points.

It is easy to see that this variant on the Euclidean travelling salesman problem is **NP**-hard as well. In this section we present an $O(\log n)$-approximation algorithm for this problem. Note that the minimum tree spanning a set of points and the minimum tour on these points are related as

$$MST(S) \leq TSP(S) \leq 2 \cdot MST(S)$$

where $MST(S)$ (resp. $TSP(S)$) denotes the minimum tree (resp. tour) spanning the points in the set S. We use this fact as follows.

Let B be the bound on the length of the tour that we are looking for and b the number of points in the optimum tour. Then the minimum tree spanning b points is of length at most B and hence our algorithm would find a tree of length at most $8\sqrt{2}\log b \cdot B$. We can now short-circuit this tree to obtain a tour of length at most twice the length of the tree. We now divide this tour into $32\sqrt{2}\log b$ segments, each having $b/(32\sqrt{2}\log b)$ vertices. Clearly one of the segments has length no more than $\frac{B}{2}$ and we can complete this segment into a tour of length at most B. Thus we have a tour of length no more than B and containing at least $b/(16\sqrt{2}\log b)$ vertices, where b is the maximum number of vertices that any tour of this length can contain. Since $b \leq n$, we obtain a $O(\log n)$-approximation to this variant of the travelling salesman problem. Since we do not the value of b in advance, we run the above procedure for all possibe choices of b (n in all) and output the best solution.

Acknowledgements We wish to acknowledge useful discussions with R. Ravi, Madhu Sudan and Randeep. We also wish to thank Vempala Santosh for suggesting the travelling salesman problem variant and H. Ramesh for the tight example in Figure 3.

References

[1] A. Aggarwal, H. Imai, N. Katoh, and S. Suri. Finding k points with minimum diameter and related problems. *Journal of Algorithms*, 12(1):38–56, 1991.

[2] Baruch Awerbuch, Yossi Azar, Avrim Blum, and Santosh Vempala. Improved approximation guarantees for minimum weight k-trees and prize-collecting salesmen. In *Proceedings, 27$^{\text{th}}$ Annual ACM Symposium on Theory of Computing*, pages 277–283, 1995.

[3] Avrim Blum, Prasad Chalasani, and Santosh Vempala. A constant-factor approximation for the k-mst problem in the plane. In *Proceedings, 27$^{\text{th}}$ Annual ACM Symposium on Theory of Computing*, pages 294–302, 1995.

[4] D. Eppstein. Faster geometric k-point mst approximation. Technical Report 13, University of California, Irvine, CA, 1995.

[5] M. Fischetti, H.W. Hamacher, K. Jornsten, and F. Maffioli. Weighted k-cardinality trees: complexity and polyhedral structure. *Networks*, 24:11–21, 1994.

[6] M. Fischetti, H.W. Hamacher, and F. Maffioli. Weighted k-cardinality trees. Technical Report 23, Politecnico di Milano, 1991.

[7] O. Goldschmidt and D.S. Hochbaum. k-edge subgraph problems. Manuscript, 1994.

[8] G. Kortsarz and D. Peleg. On choosing a dense subgraph. In *Proceedings, 34th Annual IEEE Symposium on Foundations of Computer Science*, 1993.

[9] F. Maffioli. Finding a best subtree of a tree. Technical Report 41, Politecnico di Milano, 1991.

[10] J.S.B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: a simple new method for the geometric k-mst problem. In *Proceedings, 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 402–408, 1996.

[11] V. Radhakrishnan, S. Krumke, M.V. Marathe, D.J. Rosenkrantz, and S.S. Ravi. Compact location problems. In *Proceedings, 13th Conference on Foundations of Software Technology and Theoretical computer science*, pages 238–247, 1993.

[12] R. Ravi, R. Sundaram, M.V. Marathe, D.J. Rosenkrantz, and S.S. Ravi. Spanning trees short and small. In *Proceedings, 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1993.

[13] S.S. Ravi, D.J. Rosenkrantz, and G.K. Tayi. Facility dispersion problems: Heuristics and special cases. In *Proceedings, 2nd Workshop on Algorithms and Data Structures*, 1991.