Compact Location Problems*

Sven O. Krumke[1] *Madhav V. Marathe*[2] *Venkatesh Radhakrishnan*[3]
S. S. Ravi[4] *Daniel J. Rosenkrantz*[4]

Abstract

We consider the problem of placing a specified number (p) of facilities on the nodes of a network so as to minimize some measure of the distances between facilities. This type of problem models a number of problems arising in facility location, statistical clustering, pattern recognition, and also processor allocation problems in multiprocessor systems. We consider the problem under three different objectives, namely minimizing the diameter, minimizing the average distance, and minimizing the variance. We observe that in general, the problem is NP-hard under any of the objectives. Further, even obtaining a relative approximation for any of the objectives is NP-hard.

We present a general framework for obtaining near-optimal solutions to the compact location problems for the above measures, when the distances satisfy the triangle inequality. We show that this framework can be extended to the case when there are node weights and also to the case when there is a distinguished set of nodes which must be chosen in the placement. Our approximation algorithms are easy to parallelize.

Keywords: Clustering, Location Theory, Network Design, Approximation Algorithms.

AMS(MOS) subject classification: 68Q10, 68Q15, 68Q25.

---

*A preliminary version of this paper appeared in the *Proceedings of 13th Conference on the Foundations of Software Technology and Theoretical Computer Science (FST & TCS)*, Bombay, India, Lecture Notes in Computer Science, Vol. 761, Springer-Verlag, December 1993, pp 238-247.

[1]Part of the work was done when the author was at the University at Albany. Current Address: Department of Computer Science, Universität Würzburg, Am Hubland, 97404 Würzburg, Germany. Email: `krumke@informatik.uni-wuerzberg.de`.

[2]Part of the work was done when the author was at the University at Albany. Current Address: Los Alamos National Laboratory P.O. Box 1663, MS M986, Los Alamos NM 87545. Email: `madhav@c3.lanl.gov`. The work is supported by the Department of Energy under Contract W-7405-ENG-36.

[3]Part of the work was done when the author was at the University at Albany. Current Address: Mailstop 47LA, Hewlett-Packard Company, 19447 Pruneridge Avenue, Cupertino, California 95014-9913. Email: `rven@cup.hp.com`.

[4]Department of Computer Science, University at Albany-SUNY, Albany, NY 12222. Email addresses: `{djr,ravi}@cs.albany.edu`. Supported by NSF Grants CCR-89-03319 and CCR-90-06396.

1 Introduction

We consider several problems dealing with the placement of a specified number of facilities on the nodes of a given network so as to minimize some function of the distances between the facilities. Because of the nature of the placement desired, we refer to these problems as **compact location problems**. Such location problems model several situations arising in parallel processing, statistical clustering [Ha75] and pattern recognition [An72]. For example, consider the following processor allocation problem which arises in the context of multiprocessor systems. We are given a computational task consisting of a number of communicating subtasks. At a given time, some of the processors may already be allocated and the remaining processors are available. The problem is to select a subset of processors from the currently available set of processors, one processor per subtask, such that the cost of communication among the processors executing the subtasks is minimized. In this application, the processors must be allocated quickly, and this may conflict with the goal of minimum communication cost among the selected processors.

Compact location problems also arise in a number of other applications such as allocation of manufacturing sites for the components of a system so as to minimize the cost of transporting components, distributing the activities of a project among geographically dispersed offices so as to minimize the transportation or communication cost among the offices, etc. Additional applications are discussed in [AMO93].

Such location problems are commonly modeled as problems on undirected graphs. The nodes of the graph represent the available sites. A cost is associated with each pair of sites, and it is specified as the weight of the edge joining the corresponding pair of nodes. Depending on the problem, this cost represents one of several parameters such as the cost of transporting components between the pair of sites, the cost of setting up a communication link between the pair of sites, the time required to communicate between the pair of sites, etc. In some problems, there is also a weight associated with each node. This node weight may reflect the cost of setting up a facility at the corresponding site.

Under this graph theoretic setting, a placement is a subset of nodes of a given cardinality. The cost of a placement is a problem-specific function of the weights of the nodes and edges in the subgraph induced by the placement. Examples of such cost functions are the sum of the weights of all the edges in the placement (which may reflect the total cost of setting up communication links between each pair of chosen sites), the maximum weight of an edge in the placement (i.e., the **bottleneck cost**, which may reflect the maximum time needed to communicate between any pair of chosen sites), etc. The goal of a compact location problem is to find a placement of minimum cost.

Here, we study the complexity and approximability of such compact location problems. A contribution of our paper is a general framework that leads to good approximation algorithms for compact location problems under several different objectives.

2 Summary of Results

The following is a typical compact location problem: Given an undirected graph $G = (V, E)$ with a positive distance between every pair of nodes, find a placement of p facilities such that the *diameter* (i.e., the maximum distance between any two facilities) is minimized. This problem is called the minimum diameter placement problem (MDP). Another possible objective is to minimize the *average* distance between a pair of facilities (i.e., ratio of the sum of the distances between each pair of facilities to the the number of pairs of facilities). This problem is referred to as the minimum average placement problem (MAP). Yet another objective is to minimize the *variance* of the placement (i.e., ratio of the sum of the squares of the distances between each pair of facilities to the number of pairs of facilities). This problem is referred to as the minimum variance placement problem (MVP). By a simple reduction from the CLIQUE problem [GJ79] (see Section 3), it can be shown that the problems MDP, MAP and MVP are all NP-hard, even when the distances satisfy the triangle inequality.

Given the NP-hardness of these problems, our focus is on finding good approximation algorithms for them. By an approximation algorithm (heuristic) we mean a *polynomial time* algorithm which produces feasible, but not necessarily optimal, solutions. A **relative** approximation algorithm guarantees a solution which is within a *multiplicative* constant K of the optimal value for every instance of the problem. The multiplicative constant K is referred to as the **performance guarantee** provided by the algorithm. This paper is concerned with the study of relative approximation algorithms for the placement problems defined above. It can be shown (see Section 3) that if the distances are not required to satisfy the triangle inequality, then there is no polynomial time relative approximation algorithm for any of the problems MDP, MAP and MVP, unless P = NP. This negative result prompts investigating the existence of good approximation algorithms for the problems when edge weights satisfy triangle inequality. In the past, a substantial amount of work has been carried out in investigating the approximability of problems arising in network design and location theory when edge weights satisfy the triangle inequality (for example see [RSL77, HS86, RRT94]). We refer the reader to the paper by Bern and Eppstein [BE95] for a comprehensive survey of other geometric location problems.

Here, we present a general framework for finding a near optimal placement of facilities on a given network for compact location problems. Our approximation results are summarized below.

For the MDP problem, our heuristic provides a performance guarantee of 2. We also observe that no polynomial time heuristic for the MDP problem can provide a better performance guarantee unless $\mathsf{P} = \mathsf{NP}$. For the MAP and MVP problems, our heuristics provide performance guarantees of $(2 - 2/p)$ and $(4 - 6/p)$ respectively.

We strengthen the above results above by allowing nonnegative weights on the nodes and considering minimization objectives which are functions of both node and edge weights. Specifically, we consider three minimization objectives, namely the sum of the diameter of a placement and the maximum node weight in the placement, the sum of the average edge weight in a placement and the average weight of a node in the placement, and sum of the variance of a placement and the total node weight of the placement. We denote these problems by $\mathsf{MDP}_{\mathsf{wt}}$, $\mathsf{MAP}_{\mathsf{wt}}$ and $\mathsf{MVP}_{\mathsf{wt}}$ respectively. Extending our techniques used to prove the performance guarantees for MDP, MAP and MVP, we prove that when edge weights satisfy the triangle inequality, there are polynomial time algorithms which produce placements for the problems $\mathsf{MDP}_{\mathsf{wt}}$, $\mathsf{MAP}_{\mathsf{wt}}$ and $\mathsf{MVP}_{\mathsf{wt}}$ with performance guarantees of $2 - \frac{\omega^*}{\omega^* + d^*}$, $(2 - 2/p)$ and $(4 - 6/p)$ respectively. Here, p is the number of facilities to be placed; w^*, d^* are respectively the maximum node weight and the maximum edge weight of an optimal placement.

We also extend all the above results to the case when we are given a complete graph $G(V, E)$ with node weights, edge weights satisfying triangle inequality, and also a *distinguished set* of nodes $S \subseteq V$. The problem is to find a minimum cost placement of cardinality $p + |S|$ such that there is a facility at each node in S. (The remaining p facilities must be chosen from $V - S$.) We denote the extensions of $\mathsf{MDP}_{\mathsf{wt}}$, $\mathsf{MAP}_{\mathsf{wt}}$ and $\mathsf{MVP}_{\mathsf{wt}}$ to the case with a distinguished set of nodes by $\mathsf{MDP}^{\mathsf{D}}_{\mathsf{wt}}$, $\mathsf{MAP}^{\mathsf{D}}_{\mathsf{wt}}$ and $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$ respectively. At first it may appear that these problems would be harder than the corresponding problems without distinguished nodes. However, by modifying our techniques used to prove the previous results, we show that the distinguished facilities case can be approximated with the same performance guarantees. In other words, we show that there are polynomial time algorithms for the problems $\mathsf{MDP}^{\mathsf{D}}_{\mathsf{wt}}$, $\mathsf{MAP}^{\mathsf{D}}_{\mathsf{wt}}$ and $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$ with performance guarantees of $2 - \frac{\omega^*}{\omega^* + d^*}$, $(2 - 2/(p + |S|))$ and $(4 - 6/(p + |S|))$ respectively. (The parameters ω^* and d^* have the same interpretation as before.)

Organization of the paper

The remainder of this paper is organized as follows. Section 3 contains problem definitions and preliminaries. Section 4 contains a discussion of related work. Section 5 presents our framework for heuristics and establishes performance guarantee results. Section 6 includes some geometric observations and a comparison of the three main objective functions (minimum diameter, minimum average distance and minimum variance) considered in this paper. Section 7 lists some conclusions and open problems.

In the interest of brevity, we have chosen to discuss heuristics and prove the performance guarantee results for three representative problems in this paper. They are: (i) $\mathsf{MDP_{wt}}$ (to illustrate the basic idea underlying the generic procedure), (ii) MAP (to illustrates the basic idea behind our approximation algorithms for minimizing average distance), and (iii) $\mathsf{MVP^{D}_{wt}}$ (to illustrate the ideas for minimizing variance as well as for handling distinguished nodes). We have omitted the discussion of the other variations as they readily follow from the ideas presented for one of the above three problems.

3 Preliminaries

A network is specified by a set $V = \{v_1, v_2, \ldots, v_n\}$ of nodes and a positive and symmetric distance $d(v_i, v_j)$ between every pair of distinct nodes. (We assume that $d(v_i, v_i) = 0$, $1 \le i \le n$.) Let p denote the number of facilities to be placed. A **placement** P is a subset of V containing p nodes (i.e., the nodes where facilities will be located). The **diameter** of the placement P, denoted by $\mathsf{DIA}(P)$, is given by $\mathsf{DIA}(P) = \max_{\substack{\{x,y\}, x \ne y \\ x,y \in P}} d(x,y)$. The **sum of the distances** in P, denoted by $\mathsf{SOD}(P)$, is given by $\mathsf{SOD}(P) = \sum_{\substack{\{x,y\}, x \ne y \\ x,y \in P}} d(x,y)$. The **sum of the squares of the distances** in P, denoted by $\mathsf{SSQ}(P)$, is given by $\mathsf{SSQ}(P) = \sum_{\substack{\{x,y\}, x \ne y \\ x,y \in P}} [d(x,y)]^2$. We note that the average distance and the variance[5] [AIKS91] of a placement P are equal to $\dfrac{2}{p(p-1)}\,\mathsf{SOD}(P)$ and $\dfrac{2}{p(p-1)}\,\mathsf{SSQ}(P)$ respectively.

If we allow the distances between some pairs of distinct nodes to be zero, then the optimal solution value may be zero. In such a case, the problem of obtaining a solution which is within a multiplicative factor of the optimum is trivially equivalent to the problem of finding an optimal solution. Therefore, we assume throughout this paper that for any pair of distinct nodes x and y, the distance $d(x, y)$ is strictly *positive*. We now give formal specifications of the problems considered in this paper.

I. Problems with Edge Weights only

(a) Minimum Diameter Placement (MDP)

Instance: A set $V = \{v_1, v_2, \ldots, v_n\}$ of n nodes, a positive and symmetric distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, and an integer p such that $2 \le p \le n$.

Requirement: Find a subset P of V, with $|P| = p$, such that $\mathsf{DIA}(P)$ is minimized.

(b) Minimum Average Distance Placement (MAP)

[5] In [AIKS91], the scaling factor used to define variance is $\frac{1}{p}$. We use $\frac{2}{p(p-1)}$ for reasons of uniformity.

Instance: As in MDP above.

Requirement: Find a subset P of V, with $|P| = p$, such that the average distance $\mathsf{AVG}(P)$ given by

$$\mathsf{AVG}(P) = \frac{2}{p(p-1)}\mathsf{SOD}(P)$$

is minimized.

(c) Minimum Variance Placement (MVP)

Instance: As in MDP above.

Requirement: Find a subset P of V, with $|P| = p$, such that the variance $\mathsf{VAR}(P)$ given by

$$\mathsf{VAR}(P) = \frac{2}{p(p-1)}\mathsf{SSQ}(P)$$

is minimized.

II. Edge and Node Weights

(a) Minimum Diameter Placement with Node Weights ($\mathsf{MDP}_{\mathsf{wt}}$)

Instance: A set $V = \{v_1, \ldots, v_n\}$ of n nodes with a nonnegative weight $\omega(v)$ for each $v \in V$, a positive and symmetric distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, and an integer p such that $2 \leq p \leq n$.

Requirement: Find a subset P of V, with $|P| = p$, such that

$$\mathsf{DIA}_{\mathsf{wt}}(P) = \mathsf{DIA}(P) + \max_{v \in P} \omega(v)$$

is minimized.

(b) Minimum Average Distance Placement with Node Weights ($\mathsf{MAP}_{\mathsf{wt}}$)

Instance: As in $\mathsf{MDP}_{\mathsf{wt}}$ above.

Requirement: Find a subset P of V, with $|P| = p$, such that

$$\mathsf{AVG}_{\mathsf{wt}}(P) = \frac{2}{p(p-1)}\mathsf{SOD}(P) + \sum_{v \in P} \omega(v)$$

is minimized.

(c) Minimum Variance Placement with Node Weights ($\mathsf{MVP}_{\mathsf{wt}}$)

Instance: As in $\mathsf{MDP}_{\mathsf{wt}}$ above.

Requirement: Find a subset P of V, with $|P| = p$, such that

$$\mathsf{VAR}_{\mathsf{wt}}(P) = \frac{2}{p(p-1)}\mathsf{SSQ}(P) + \sum_{v \in P} \omega(v)$$

is minimized.

III. Edge Weights and Distinguished Nodes

(a) Minimum Diameter Placement with Distinguished Nodes ($\mathsf{MDP}^{\mathsf{D}}$)

Instance: A set $V = \{v_1, \ldots, v_n\}$ of n nodes with a positive and symmetric distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, a subset $S \subseteq V$, and an integer p such that $0 \leq p \leq |V - S|$.

Requirement: Find a subset P of $V - S$, with $|P| = p$, such that $\mathsf{DIA}^{\mathsf{D}}(P) = \mathsf{DIA}(P \cup S)$ is minimized.

(b) Minimum Average Distance Placement with Distinguished Nodes ($\mathsf{MAP}^{\mathsf{D}}$)

Instance: As in $\mathsf{MDP}^{\mathsf{D}}$ above.

Requirement: Find a subset P of $V - S$, with $|P| = p$, such that

$$\mathsf{AVG}^{\mathsf{D}}(P) = \frac{2}{(p + |S|)(p + |S| - 1)}\mathsf{SOD}(P \cup S)$$

is minimized.

(c) Minimum Variance Placement with Distinguished Nodes ($\mathsf{MVP}^{\mathsf{D}}$)

Instance: As in $\mathsf{MDP}^{\mathsf{D}}$ above.

Requirement: Find a subset P of $V - S$, with $|P| = p$, such that

$$\mathsf{VAR}^{\mathsf{D}}(P) = \frac{2}{(p + |S|)(p + |S| - 1)}\mathsf{SSQ}(P \cup S)$$

is minimized.

IV. Edge Weights, Node Weights and Distinguished Nodes

(a) Minimum Diameter Placement with Node Weights and Distinguished Nodes ($\mathsf{MDP}^{\mathsf{D}}_{\mathsf{wt}}$)

Instance: A set $V = \{v_1, \ldots, v_n\}$ of n nodes with a nonnegative weight $\omega(v)$ for each $v \in V$, a positive and symmetric distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, a subset $S \subseteq V$, and an integer p such that $0 \leq p \leq |V - S|$.

Requirement: Find a subset P of $V - S$, with $|P| = p$, such that

$$\mathsf{DIA}^{\mathsf{D}}_{\mathsf{wt}}(P) = \mathsf{DIA}(P \cup S) + \max_{v \in P \cup S} \omega(v)$$

is minimized.

(b) Minimum Average Distance Placement with Node Weights and Distinguished Nodes ($\mathsf{MAP}^{\mathsf{D}}_{\mathsf{wt}}$)

Instance: As in $\mathsf{MDP}^{\mathsf{D}}_{\mathsf{wt}}$ above.

Requirement: Find a subset P of $V - S$, with $|P| = p$, such that

$$\mathsf{AVG}^{\mathsf{D}}_{\mathsf{wt}}(P) = \frac{2}{(p + |S|)(p + |S| - 1)} \mathsf{SOD}(P \cup S) + \sum_{v \in P \cup S} \omega(v)$$

is minimized.

(c) Minimum Variance Placement with Node Weights and Distinguished Nodes ($\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$)

Instance: As in $\mathsf{MDP}^{\mathsf{D}}_{\mathsf{wt}}$ above.

Requirement: Find a subset P of $V - S$, with $|P| = p$, such that

$$\mathsf{VAR}^{\mathsf{D}}_{\mathsf{wt}}(P) = \frac{2}{(p + |S|)(p + |S| - 1)} \mathsf{SSQ}(P \cup S) + \sum_{v \in P \cup S} \omega(v)$$

is minimized.

We assume throughout this paper that the distances specified in the instances of the above problems satisfy the **triangle inequality**. That is, for any three distinct nodes v_i, v_j and v_k, $d(v_i, v_j) + d(v_j, v_k) \geq d(v_i, v_k)$.

We end this section with two propositions. The first proposition establishes the NP-hardness of MDP, MAP and MVP problems even when the distances satisfy the triangle inequality. The second proposition points out that unless P = NP, there is no relative approximation algorithm for any of these problems when the distances are not required to satisfy the triangle inequality.

Proposition 3.1 The problems MDP, MAP and MVP are NP-hard, even when the distances satisfy the triangle inequality.

Proof: We will present the proof NP-hardness for the MDP problem. The proofs for the other problems are similar.

We use a reduction from CLIQUE. Consider an instance of the CLIQUE problem consisting of the undirected graph $G(V, E)$ and an integer $J \leq |V|$. Construct an instance of MDP consisting of the complete graph on the node set V. For nodes x and y in V, let $d(x, y) = 1$ if $\{x, y\} \in E$ and let $d(x, y) = 2$ otherwise. Clearly, the distances satisfy the triangle inequality. It is straightforward to verify that G has a clique of size J if and only if we can place J facilities such that the diameter of the placement is equal to 1. ■

Proposition 3.2 If the distances are not required to satisfy the triangle inequality, then there is no polynomial time relative approximation algorithm for any of the problems MDP, MAP and MVP unless P = NP.

Proof: Here also, we will present the proof for the MDP problem. Suppose there is a polynomial time approximation algorithm $\mathcal{A}$ which provides performance guarantee of K for the MDP problem. We use the construction discussed in the proof of Proposition 3.1 except that $d(x, y)$ is set to $K + 1$ when $\{x, y\}$ is not in E. As before, if G has a clique with J nodes, then the diameter of an optimal placement of size J is 1. If G does not have a clique with J nodes, then the diameter any placement of size J is $K + 1$. Therefore, approximation algorithm $\mathcal{A}$ produces a placement with diameter 1 if and only if G ha a clique of size J. Since $\mathcal{A}$ runs in polynomial time, this contradicts the assumption that P $\neq$ NP. ■

4 Related Work

In contrast to the above NP-hardness results which hold for general distance matrices, geometric versions of MDP and MVP were shown to be solvable in polynomial time in [AIKS91]. In the geometric versions of these problems, the nodes are points in space and the distance between a pair of nodes is their Euclidean distance. For points in the plane, [AIKS91] contains an $O(p^{2.5} n \log p + n \log n)$ algorithm for the MDP problem and an $O(p^2 n \log n)$ algorithm for the MVP problem, and it is observed that these algorithms extend to higher dimensions. These algorithms are based on the construction of p^{th} order Voronoi diagrams [Le82, PS85].

Other work has addressed placement problems where the objective functions are different from the above. For example, traditional facility location problems are concerned with minimizing the maximum distance from a node to a nearest facility (p-center problem) or minimizing the sum of the distances from each node to the nearest facility (p-median problem) [HM79, MF90]. However, for the problems considered in this paper, the objective functions involve only the distances between facilities. The *clustering* problems considered in the literature [HS86, FG88, Go85] involve partitioning a given set of nodes into clusters so as to minimize a specified objective function. Thus, these problems are different from the compact location problems studied in this paper. Recently, several researchers have reported results on a compact location problem with a different objective function [RS+94, GH94, AA+95, BC+95].

procedure $\textsf{Gen-Alg}^{\textsf{D}}_{\textsf{wt}}(G(V, E), S, F)$

Comment: $G(V, E)$ denotes the graph with edge and/or node weights. S is the distinguished set of nodes. F is a function that returns the cost of a given placement.

1. **if** $|V - S| < p$ **then return** "certificate of failure"
2. **else**
   begin
   (a) Solution $\leftarrow \emptyset$ and Value $\leftarrow \infty$.
   (b) **for** each node $v \in V$ **do**
   begin
   i. **if** $v \in S$ **then** $k \leftarrow p$ **else** $k \leftarrow p - 1$.
   ii. Find $P_v = \{v_1, \ldots, v_k\} \subseteq V - S - \{v\}$ such that P_v contains $p - 1$ nodes in $V - S - \{v\}$ closest to v.
   iii. **if** $v \notin S$ **then** $P_v = P_v \cup \{v\}$.
   iv. $t \leftarrow F(P_v)$.
   v. **if** $t <$ Value **then** Solution $\leftarrow P_v$ and Value $\leftarrow t$.
   end
   end
3. **output** Solution and Value.

Figure 1: Generic Algorithm

We refer to that problem as the p-MST problem since the cost of a placement with p nodes is the weight of a minimum spanning tree on the p nodes. The problem was defined in [RS+94]. There, it is shown that even the geometric version of the problem is NP-hard. References [GH94, BC+95] present approximation algorithms for the geometric version. An approximation algorithm for the general version of the problem appears in [AA+95].

Facility Location problems where the objective is to place facilities so as to *maximize* some function of the distances between facilities have also been considered in the literature; see for example [EN89, Ta91, KP93, RRT94]. Variations of such maximization problems in which the placement of facilities is not restricted to the nodes of the network are discussed in [MF90]. We consider only location problems in which the facilities are placed at the nodes of the network.

We also consider problems where the goal is to minimize some function of node weights and edge weights. Such objective functions have been considered in the context of other location problems. For example, Hochbaum and Shmoys [HS86] and Dyer and Frieze [DF85] consider the node weighted versions of the p-center problem. Location problems with other optimization criteria have also been considered in the literature. For example, Lin and Vitter

[LV92] provide approximation algorithms for the p-median problem. The solution produced is approximate in terms of both the number of median nodes used and the sum of the distances from each node to the nearest median. Bar-Ilan, Kortsarz and Peleg [BKP93] consider the balanced center problem. They provide approximation algorithms for problem of allocating network centers where each center is allowed to service at most a specified number of nodes.

5 Approximation Results

5.1 Generic Procedure

In this section we discuss our approximation algorithms and establish their performance guarantees. Throughout this section we assume that the edge weights satisfy the triangle inequality. All our approximation algorithms make use of the generic procedure given in Figure 1. As can be seen, the generic algorithm has three parameters, namely the graph instance $G(V, E)$ with node and edge weights, the set S of distinguished nodes, and a function F that maps subsets of V of cardinality $p+|S|$ (i.e., feasible placements) into $\mathbb{R}^+$ (i.e., costs of placements), depending on the problem instance.

5.2 Approximating the Diameter Plus the Maximum Node Weight

Here, we consider the $\mathsf{MDP}_{\mathsf{wt}}$ problem. Our approximation algorithm for this problem, denoted by Heur-$\mathsf{MDP}_{\mathsf{wt}}$, is shown in Figure 2. The algorithm is reminiscent of the generic procedure given for other bottleneck problems by Hochbaum and Shmoys [HS86]. It returns a placement of cost ∞ if a feasible solution is not found. We now define the concept of a bottleneck subgraph (BOTT) used in the description of the algorithm.

Definition 5.1 Let $G(V, E)$ be a complete graph with node set $V = \{v_1, \cdots, v_n\}$, a positive weight $d(v_i, v_j)$ for each edge $\{v_i, v_j\}$ and a nonnegative weight $\omega(v)$ for each node $v \in V$. For any $c \in \mathbb{R}^+$, we define **the bottleneck subgraph** of G, denoted by $\mathsf{BOTT}(V, c)$, to be the complete subgraph induced by the set of nodes $V' = \{v \in V \,|\, \omega(v) \leq c\}$. The weight of any edge in $\mathsf{BOTT}(V, c)$ is equal to its weight in G; the weight of any node in $\mathsf{BOTT}(V, c)$ is equal to its weight in G.

We are now ready to establish the performance guarantee provided by Heur-$\mathsf{MDP}_{\mathsf{wt}}$.

Theorem 5.2 Let I be an instance of $\mathsf{MDP}_{\mathsf{wt}}$. Let $P^* \subseteq V$ be an optimal placement and let $P \subseteq V$ be the placement produced by Heur-$\mathsf{MDP}_{\mathsf{wt}}$. Then $\mathsf{DIA}_{\mathsf{wt}}(P)/\mathsf{DIA}_{\mathsf{wt}}(P^*) \leq \left(2 - \frac{\omega^*}{d^* + \omega^*}\right)$, where d^* is the maximum distance between two nodes in P^* and ω^* is the maximum node weight in P^*.

procedure $\mathsf{Heur\text{-}MDP_{wt}}$

1. Solution $\leftarrow \emptyset$ and Value $\leftarrow \infty$.
2. Let $\omega_1, \omega_2, \ldots, \omega_n$ denote the the weights of nodes $v_1, v_2, \ldots, v_n$ respectively.
3. **for** $i := 1$ **to** n **do**

   begin

   (a) $W \leftarrow \mathsf{BOTT}(V, \omega_i)$.

   (b) Run $\mathsf{Gen\text{-}Alg^D_{wt}}$ with $G = W$, $S = \emptyset$ and $F = \mathsf{DIA}$. Let the result be P.

   (c) **if** P is not a "certificate of failure" **then** $t \leftarrow \mathsf{DIA_{wt}}(P)$ **else** $t \leftarrow \infty$.

   (d) **if** $t <$ Value **then** Value $\leftarrow t$ and Solution $\leftarrow P$.

   end
4. **output** Solution and Value.

Figure 2: Heuristic for the $\mathsf{MDP_{wt}}$ Problem

Proof: We look at the iteration where $\mathsf{Heur\text{-}MDP}_{wt}$ considers $\mathsf{BOTT}(V, \omega^*)$. By the definition of ω^*, it follows that all the nodes of P^* appear in $\mathsf{BOTT}(V, \omega^*)$. Consequently, $\mathsf{Gen\text{-}Alg}_{wt}$ will find a solution for $\mathsf{BOTT}(V, \omega^*)$, and will not return a "certificate of failure".

Let $v \in P^*$ be an arbitrary node and consider the iteration of $\mathsf{Gen\text{-}Alg^D_{wt}}$, where v is considered. Let $P_v = \{w_1, \cdots, w_{p-1}\}$ be the set of nearest neighbors of v chosen in that iteration. By the definition of $\mathsf{BOTT}(V, \omega^*)$, it follows that

$$\omega(w_i) \leq \omega^*, \quad (i = 1, \ldots, p-1). \tag{1}$$

The set P_v is chosen to be the set of nearest neighbors of v. Consequently, we have

$$d(v, w_i) \leq d^*, \quad (i = 1, \ldots, p-1). \tag{2}$$

Let $u, w \in P_v$ be arbitrary nodes. By using the triangle inequality and Inequality (2) it follows that

$$d(u, w) \leq d(u, v) + d(v, w) \leq d^* + d^* = 2d^*. \tag{3}$$

Therefore, by Inequalities (1) and (2) we have

$$\mathsf{DIA_{wt}}(P_v) \leq 2d^* + \omega^* = \left(2 - \frac{\omega^*}{d^* + \omega^*}\right)(d^* + \omega^*) = \left(2 - \frac{\omega^*}{d^* + \omega^*}\right)\mathsf{DIA_{wt}}(P^*). \tag{4}$$

Since $\mathsf{Heur\text{-}MDP_{wt}}$ picks a placement P which has the minimum cost among all the placements considered, it follows that $\mathsf{DIA_{wt}}(P) \leq \mathsf{DIA_{wt}}(P_v) \leq \left(2 - \frac{\omega^*}{d^* + \omega^*}\right)\mathsf{DIA_{wt}}(P^*)$. ■

From Theorem 5.2 it can be seen that when the node weights are zero (i.e., when we consider the MDP problem), the performance guarantee provided by the heuristic is 2. The following proposition points out that if $\mathsf{P} \neq \mathsf{NP}$, this performance guarantee for the MDP problem cannot be improved by any polynomial time algorithm.

Proposition 5.3 If $\mathsf{P} \neq \mathsf{NP}$, then no polynomial time approximation algorithm can provide a performance guarantee of $(2-\epsilon)$ for any $\epsilon > 0$ for the MDP problem.

Proof: We use the reduction from CLIQUE presented in the proof of Proposition 3.1. If G has a clique of size J, then we can place J facilities such that the diameter of the placement is equal to 1. Otherwise, every placement of J facilities has a diameter of 2. Thus, any polynomial approximation algorithm that provides a performance guarantee of $2-\epsilon$ for any $\epsilon > 0$ can be used to solve the CLIQUE problem in polynomial time, contradicting the assumption that $\mathsf{P} \neq \mathsf{NP}$. ■

We now point out how Heur-$\mathsf{MDP}_{\mathsf{wt}}$ can be parallelized.

Proposition 5.4 Heur-$\mathsf{MDP}_{\mathsf{wt}}$ can be implemented to run in $O(\log n)$ parallel time using $O(\max\{n^3, n^2p^2/\log p\})$ processors.

Proof: An examination of Figure 2 shows that Heur-$\mathsf{MDP}_{\mathsf{wt}}$ constructs each of the n bottleneck subgraphs of G, executes the generic procedure on each bottleneck subgraph with $S = \emptyset$ and $F = \mathsf{DIA}$, and returns the best placement among those produced by the generic procedure. In a parallel implementation, the n bottleneck subgraphs are examined in parallel.

We first determine the time and processor bounds for a parallel implementation of the generic procedure (Figure 1) with $S = \emptyset$ and $F = \mathsf{DIA}$. The basic idea is to implement the **for** loop of (Figure 1) as a parallel step over all nodes. For each node v, the set P_v can be computed in $O(\log n)$ time using $O(n)$ processors by sorting the nodes in non-decreasing order of their distances from v [Ja92]. Using standard techniques [Ja92], for each set P_v, the diameter can be computed in $O(\log p)$ time using $O(p^2/\log p)$ processors, since this entails merely the computation of the maximum value of the $p(p-1)/2$ edge distances in P_v. The solution set can be obtained in $O(\log n)$ time using $O(n/\log n)$ processors by computing the minimum diameter among the sets P_v, for all $v \in V$. Since this sequence of operations is done in parallel for all nodes, the generic procedure can be implemented to run in $O(\log n)$ time using $O(\max\{n^2, np^2/\log p\})$ processors.

Let us now consider Heur-$\mathsf{MDP}_{\mathsf{wt}}$. Each bottleneck subgraph can be constructed in $O(1)$ parallel time using $O(n^2)$ processors. As discussed above, for each bottleneck subgraph, the generic procedure runs in $O(\log n)$ time using $O(\max\{n^2, np^2/\log p\})$ processors. Thus, using $O(\max\{n^3, n^2p^2/\log p\})$ processors, the solution values for all the bottleneck subgraphs can be obtained in $O(\log n)$ time. Finally, a minimum cost solution can be obtained in $O(\log n)$ time using $O(n/\log n)$ processors.

Thus, the overall parallel time is $O(\log n)$ and the number of processors used is $O(\max\{n^3, n^2p^2/\log p\})$. ■

5.3 Results for Average Distance

We now consider the problem MAP where the goal is to find a placement P for which the average distance $\mathsf{AVG}(P)$ given by

$$\mathsf{AVG}(P) = \frac{2}{p(p-1)}\mathsf{SOD}(P)$$

is minimized. We note that obtaining a placement P for which $\mathsf{AVG}(P)$ is a minimum is equivalent to obtaining a placement with the minimum value of $\mathsf{SOD}(P)$. Further, for any two placements P and Q, the ratio $\mathsf{SOD}(P)/\mathsf{SOD}(Q)$ is equal to the ratio $\mathsf{AVG}(P)/\mathsf{AVG}(Q)$. Therefore, in determining the performance guarantee provided by a heuristic for the MAP problem, we can ignore the scaling factor $2/(p(p-1))$.

Our heuristic for the MAP problem, denoted by Heur-MAP, consists merely of a call to the generic procedure of Figure 1 with G as the graph corresponding to the problem instance, $S = \emptyset$ and $F = \mathsf{SOD}$. The following theorem establishes the performance guarantee provided by this heuristic.

Theorem 5.5 Let I be an instance of MAP. Let P^* and P denote respectively an optimal placement and that produced by Heur-MAP for the instance I. Then $\mathsf{SOD}(P)/\mathsf{SOD}(P^*) \leq (2 - 2/p)$.

Proof: For each node $w \in P^*$, let

$$R_w \quad = \quad \sum_{u \in P^*} d(w, u).$$

Then

$$\mathsf{SOD}(P^*) \quad = \quad \frac{1}{2} \sum_{w \in P^*} \sum_{u \in P^*} d(u, w) \quad = \quad \frac{1}{2} \sum_{w \in P^*} R_w.$$

Let v be a node in P^* such that $R_v \leq R_w$ for all $w \in P^*$, Therefore,

$$\mathsf{SOD}(P^*) \quad = \quad \frac{1}{2} \sum_{w \in P^*} R_w \quad \geq \quad \frac{1}{2} \sum_{w \in P^*} R_v \quad = \quad \frac{p}{2} R_v.$$

Consequently,

$$R_v \quad \leq \quad \frac{2}{p}\, \mathsf{SOD}(P^*). \tag{5}$$

Recall that Heur-MAP consists simply of an execution of the generic procedure with $S = \emptyset$ and $F = \mathsf{SOD}$. Consider the iteration of the for loop (Step 2(b) of Figure 1) in which the vertex v chosen above is considered. Let P_v be the placement chosen in Step 2(b)(iii) of this iteration. Since P_v is one of the placements considered by the heuristic and the heuristic outputs a subset with the minimum sum of distances among all such subsets, we observe that

$$\mathsf{SOD}(P) \leq \mathsf{SOD}(P_v). \tag{6}$$

Define

$$Q_v = \sum_{w \in P_v - \{v\}} d(v, w).$$

Note that

$$\mathsf{SOD}(P_v) = \sum_{w \in P_v - \{v\}} d(v, w) + \sum_{\substack{\{a,b\}, a \neq b \\ a,b \in P_v - \{v\}}} d(a, b)$$

The first term on the right side of the above equation is equal to Q_v. To bound the second term, we note that by the triangle inequality, $d(a, b) \leq d(a, v) + d(v, b)$. Since the distances are symmetric, $d(a, v) = d(v, a)$. Thus, $d(a, b) \leq d(v, a) + d(v, b)$. Consequently,

$$\begin{aligned}
\sum_{\substack{\{a,b\}, a \neq b \\ a,b \in P_v - \{v\}}} d(a, b) &\leq \sum_{\substack{\{a,b\}, a \neq b \\ a,b \in P_v - \{v\}}} (d(v, a) + d(v, b)) \\
&= \sum_{c \in P_v - \{v\}} \sum_{d \in P_v - \{v\}} d(v, c) \\
&= (p - 2) \sum_{c \in P_v - \{v\}} d(v, c) = (p - 2)Q_v
\end{aligned}$$

Therefore,

$$\mathsf{SOD}(P_v) \leq Q_v + (p - 2)Q_v = (p - 1)Q_v \tag{7}$$

Also note that $Q_v \leq R_v$ because Q_v is the sum of the distances to the $p - 1$ nodes nearest to v and R_v is the sum of the distances from v to the other $p - 1$ nodes in P^*. Therefore from (5), we obtain

$$Q_v \leq \frac{2}{p} \mathsf{SOD}(P^*). \tag{8}$$

Now, from (7) and (8), $\mathsf{SOD}(P_v) \leq (2 - 2/p)\, \mathsf{SOD}(P^*)$. Hence, from (6), $\mathsf{SOD}(P)/\mathsf{SOD}(P^*) \leq (2 - 2/p)$. ∎

Lower bound example: Consider the following instance I of the MAP problem. The set of nodes $V = A \cup B_1 \cup \cdots \cup B_p$ where, $A = \{a_1, \cdots, a_p\}$ and $B_i = \{b_{i,1}, \cdots, b_{i,p-1}\}$, $1 \leq i \leq p$. We define the distances between the nodes as follows.

$d(a_i, a_j) = 1$ for $1 \leq i \leq p$, $1 \leq j \leq p$, and $i \neq j$.
$d(a_i, b_{i,j}) = 1 - \epsilon$ for $1 \leq i \leq p$, $1 \leq j \leq p - 1$, .
$d(a_i, b_{j,k}) = 2 - \epsilon$ for $1 \leq i \leq p$, $1 \leq j \leq p$, $1 \leq k \leq p - 1$, and $i \neq j$.
$d(b_{i,j}, b_{i,k}) = 2 - 2\epsilon$ for $1 \leq i \leq p$, $1 \leq j \leq p - 1$, $1 \leq k \leq p - 1$, and $j \neq k$.
$d(b_{i,j}, b_{s,t}) = 3 - 2\epsilon$ for $1 \leq i \leq p$, $1 \leq j \leq p - 1$, $1 \leq s \leq p$, $1 \leq t \leq p - 1$ and $i \neq s$.

Note that the distances specified above satisfy the triangle inequality. An optimal set of p nodes with respect to sum of the distances is the set A, for which $\mathsf{SOD}(A) = p(p - 1)/2$. For any point x, let S_x be the set of p nodes closest to x. It can be verified that $S_{a_i} = \{a_i\} \cup B_i$

and $S_{b_{i,j}} = \{a_i\} \cup B_i$. It can also be verified that all these sets have the same value with respect to the SOD function. So assume that $\mathsf{Heur\text{-}MAP}$ chooses S_{a_1} as the placement. Now, $\mathsf{SOD}(S_{a_1}) = (p-1)(1-\epsilon) + (p-1)(p-2)(2-2\epsilon)/2$. Now, as $\epsilon \to 0$, the ratio $\mathsf{SOD}(S_{a_1})/\mathsf{SOD}(A)$ approaches $2 - 2/p$.

Since $\mathsf{Heur\text{-}MAP}$ involves just one execution of the generic procedure with $S = \emptyset$ and $F = \mathsf{SOD}$, a parallel implementation of $\mathsf{Heur\text{-}MAP}$ runs in $O(\log n)$ time and $O(\max\{n^2, np^2/\log p\})$ processors, as explained in the proof of Proposition 5.4.

5.4 Approximating Variance Plus Sum of Node Weights with Distinguished Nodes

Here, we consider the problem $\mathsf{MAP}^{\mathsf{D}}_{\mathsf{wt}}$. Recall that the goal of this problem is to find a placement P for which the value $\mathsf{VAR}^{\mathsf{D}}_{\mathsf{wt}}(P)$ given by

$$\mathsf{VAR}^{\mathsf{D}}_{\mathsf{wt}}(P) = \frac{2}{(p+|S|)(p+|S|-1)}\mathsf{SSQ}(P \cup S) + \sum_{v \in P \cup S} \omega(v)$$

is minimized.

We obtain an approximation algorithm for the $\mathsf{MAP}^{\mathsf{D}}_{\mathsf{wt}}$ problem in two steps. First, we transform the given instance of the $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$ problem into an instance of the $\mathsf{MVP}^{\mathsf{D}}$ problem. This transformation has the property that any solution to the resulting $\mathsf{MVP}^{\mathsf{D}}$ instance with performance guarantee α implies a solution with the same performance guarantee α for the original instance of the $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$ problem. In the second step, we present an approximation algorithm for the $\mathsf{MVP}^{\mathsf{D}}$ problem with a performance guarantee of $(4 - \frac{6}{p+|S|})$. In view of the approximation-preserving nature of our transformation, this performance guarantee applies to the $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$ problem as well. We begin with the transformation and a proof of its correctness.

Theorem 5.6 Given an instance I of $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$, we can construct an instance I' of $\mathsf{MVP}^{\mathsf{D}}$ in polynomial time with the following property. Given a placement for the instance I' of the $\mathsf{MVP}^{\mathsf{D}}$ problem with performance guarantee α, the same placement provides a solution to the instance I of the $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$ problem with the same performance guarantee α.

Proof: Given an instance I of $\mathsf{MVP}^{\mathsf{D}}_{\mathsf{wt}}$, we construct an instance I' of $\mathsf{MVP}^{\mathsf{D}}$ as follows. Let $V' = V$, $S' = S$, $p' = p$, and for $u, v \in V$, let

$$d'(u,v) = \begin{cases} \sqrt{d(u,v)^2 + \frac{1}{2}(p+|S|)(\omega(v)+\omega(w))} & \text{if } u \neq v \\ 0 & \text{otherwise} \end{cases}$$

We will first show that the triangle inequality holds in I'. Let $\lambda = (p+|S|)/2$. Then for arbitrary nodes $u, v, w \in V'$

$$(d'(u,v) + d'(v,w))^2 \quad = \quad d(u,v)^2 + d(v,w)^2 + \lambda(\omega(u) + 2\omega(v) + \omega(w)) +$$

$$
\begin{aligned}
& +2\sqrt{(d(u,v)^2+\lambda(\omega(u)+\omega(v)))(d(v,w)^2+\lambda(\omega(v)+\omega(w)))} \\
\geq\ & d(u,v)^2+d(v,w)^2+\lambda(\omega(u)+2\omega(v)+\omega(w))+2d(u,v)d(v,w) \\
=\ & (d(u,v)+d(v,w))^2+\lambda(\omega(u)+2\omega(v)+\omega(w)) \\
\geq\ & d(u,w)^2+\lambda(\omega(u)+\omega(w))=d'(u,w)^2.
\end{aligned}
$$

Therefore, $d'(u,v)+d'(v,w)\geq d'(u,w)$, as required by the triangle inequality.

We now prove that for any feasible placement $P\subseteq V-S$, the values of $\mathsf{VAR}^{\mathsf{D}}_{\mathsf{wt}}(P)$ and $\mathsf{VAR}^{\mathsf{D}}(P)$ are equal, where $\mathsf{VAR}^{\mathsf{D}}_{\mathsf{wt}}(P)$ uses the distance function d and $\mathsf{VAR}^{\mathsf{D}}(P)$ uses the distance function d'.

Let $P\subseteq V-S$ and $|P|=p$. Let $X=\sum\limits_{r,q\in S}[d(r,q)]^2$ and $Y=\sum\limits_{r\in S}\omega(r)$. We have

$$
\begin{aligned}
\mathsf{VAR}^{\mathsf{D}}(P) &= \frac{2}{(p+|S|)(p+|S|-1)}\mathsf{SSQ}(P\cup S) \\
&= \frac{2}{(p+|S|)(p+|S|-1)}\sum_{v,w\in P\cup S, v\neq w}[d'(v,w)]^2 \\
&= \frac{2}{(p+|S|)(p+|S|-1)}\sum_{v,w\in S, v\neq w}[d'(v,w)]^2+ \\
&\quad +\frac{2}{(p+|S|)(p+|S|-1)}\sum_{v\in P, s\in S}[d'(v,s)]^2+ \\
&\quad +\frac{2}{(p+|S|)(p+|S|-1)}\sum_{v,w\in P, v\neq w}[d'(v,w)]^2
\end{aligned}
$$

We consider the three sums in the last equality separately. The first sum can be expressed as follows.

$$
\begin{aligned}
\sum_{v,w\in S,v\neq w}[d'(v,w)]^2 &= \sum_{v,w\in S,v\neq w}[d(v,w)]^2+\frac{1}{2}(p+|S|)(\omega(v)+\omega(w)) \\
&= X+\frac{1}{2}(p+|S|)(|S|-1)\sum_{v\in S}\omega(v) \\
&= X+\frac{1}{2}(p+|S|)(|S|-1)Y.
\end{aligned}
$$

The second sum is treated as follows.

$$
\begin{aligned}
\sum_{v\in P,s\in S}[d'(v,s)]^2 &= \sum_{v\in P,s\in S}[d(v,s)]^2+\frac{1}{2}(p+|S|)(\omega(v)+\omega(s)) \\
&= \sum_{v\in P,s\in S}[d(v,s)]^2+\frac{1}{2}(p+|S|)pY+\frac{1}{2}(p+|S|)|S|\sum_{v\in P}\omega(v).
\end{aligned}
$$

Finally, the third sum equals

$$\sum_{v,w\in P, v\neq w} [d'(v,w)]^2 = \sum_{v,w\in P, v\neq w} [d(v,w)]^2 + \frac{1}{2}(p+|S|)(\omega(v)+\omega(s))$$
$$= \sum_{v,w\in P} [d(v,w)]^2 + \frac{1}{2}(p+|S|)(p-1)\sum_{v\in P}\omega(v).$$

Putting together these results yields

$$\begin{aligned}\mathsf{VAR}^{\mathsf{D}}(P) &= \frac{2}{(p+|S|)(p+|S|-1)}\Bigg(X + \sum_{v\in P, s\in S}[d(v,s)]^2 + \sum_{v,w\in P}[d(v,w)]^2 + \\ &\quad + \frac{1}{2}(p+|S|)(p+|S|-1)\left(Y + \sum_{v\in P}\omega(v)\right)\Bigg) \\ &= \mathsf{VAR}^{\mathsf{D}}_{\mathsf{wt}}(P).\end{aligned}$$

Thus, feasible placements for I are exactly the feasible placements for I'. It follows that a placement P^* is optimal for I if and only if P^* is optimal for I', and a placement P has a performance guarantee of α with respect to the optimal for I if and only if P has a performance guarantee of α with respect to the optimal for I'. ■

Having completed the transformation, we focus on the approximation algorithm for $\mathsf{MVP}^{\mathsf{D}}$. In this problem, we are interested in finding a placement P for which the value $\mathsf{VAR}^{\mathsf{D}}(P)$ given by

$$\mathsf{VAR}^{\mathsf{D}}(P) = \frac{2}{(p+|S|)(p+|S|-1)}\mathsf{SSQ}(P\cup S)$$

is minimized. Our approximation algorithm for this problem, denoted by $\mathsf{Heur\text{-}MVP}^{\mathsf{D}}$, consists merely of a call to the generic procedure outlined in Figure 1 with G and S as given in the problem instance and the function F defined by $F(P) = \mathsf{SSQ}(P\cup S)$. As in the case of the $\mathsf{MAP}_{\mathsf{wt}}$ problem, for any subsets P and Q of $V - S$ with $|P| = |Q| = p$, the ratio $\mathsf{VAR}^{\mathsf{D}}(P)/\mathsf{VAR}^{\mathsf{D}}(Q)$ is equal to the ratio $\mathsf{SSQ}(P\cup S)/\mathsf{SSQ}(Q\cup S)$. Therefore, we state and prove the performance guarantee result for $\mathsf{Heur\text{-}MVP}^{\mathsf{D}}$ in terms of the ratios of the SSQ function values. This proof requires an extended form of the triangle inequality given by the following lemma.

Lemma 5.7 If the distances in a network satisfy the triangle inequality, then for all nodes x, y, z, $[d(x,z)]^2 \le 2\left([d(x,y)]^2 + [d(y,z)]^2\right)$.

Proof: By the triangle inequality, we have $d(x,z) \le d(x,y) + d(y,z)$. By squaring both sides we obtain

$$[d(x,z)]^2 \le [d(x,y)]^2 + [d(y,z)]^2 + 2d(x,y)d(y,z). \tag{9}$$

Moreover, we obtain from

$$0 \le (d(x,y) - d(y,z))^2 = [d(x,y)]^2 + [d(y,z)]^2 - 2d(x,y)d(y,z) \tag{10}$$

that

$$2d(x,y)d(y,z) \le [d(x,y)]^2 + [d(y,z)]^2 . \tag{11}$$

Thus by (9) and (11)

$$[d(x,z)]^2 \le [d(x,y)]^2 + [d(y,z)]^2 + 2d(x,y)d(y,z) \le 2\left([d(x,y)]^2 + [d(y,z)]^2\right) \tag{12}$$

which proves the lemma. ■

Theorem 5.8 Let I be an instance of $\mathsf{MVP}^{\mathsf{D}}$. Let $P^* \subseteq V - S$ be an optimal placement and let $P \subseteq V - S$ be the placement produced by Heur-$\mathsf{MVP}^{\mathsf{D}}$ respectively for the instance I. Then $\mathsf{SSQ}(P \cup S)/\mathsf{SSQ}(P^* \cup S) \le 4 - 6/(p + |S|)$.

Proof: For each node $w \in P^* \cup S$, let

$$R_w = \sum_{v \in P^* \cup S} [d(v,w)]^2 . \tag{13}$$

We have

$$\mathsf{SSQ}(P^* \cup S) = \frac{1}{2} \sum_{w \in P^* \cup S} R_w. \tag{14}$$

Choose $v \in S \cup P^*$ such that

$$R_v = \min_{w \in P^* \cup S} R_w. \tag{15}$$

Then

$$\mathsf{SSQ}(P^* \cup S) = \frac{1}{2} \sum_{w \in P^* \cup S} R_w \ge \frac{1}{2} \sum_{w \in P^* \cup S} R_v \quad = \quad \frac{p + |S|}{2} R_v. \tag{16}$$

This yields

$$R_v \le \frac{2}{p + |S|} \mathsf{SSQ}(P^* \cup S). \tag{17}$$

Consider the iteration of the for loop (Step 2(b) of Figure 1), where the node v chosen above is considered by the heuristic. Let $P_v = \{w_1, \ldots, w_k\}$ be the set of nearest neighbors of v in $V - S - \{v\}$ chosen in that iteration, where $k = p$ if $v \in S$ and $k = p - 1$ if $v \in P^*$. Define

$$Q_v = \sum_{w \in P_v \cup S - \{v\}} [d(v,w)]^2 . \tag{18}$$

By definition of R_v and Q_v we have

$$Q_v = \sum_{s \in S} [d(v,s)]^2 + \sum_{w \in P_v} [d(v,w)]^2 \le \sum_{s \in S} [d(v,s)]^2 + \sum_{w \in P^*} [d(v,w)]^2 = R_v, \tag{19}$$

because P_v is chosen as the set of k nearest neighbors of v.

We prove the theorem by showing that

$$\mathsf{SSQ}(P_v \cup S) \leq [4 - 6/(p + |S|)]\mathsf{SSQ}(P^* \cup S). \tag{20}$$

This can be proven as follows.

$$\begin{aligned}
\mathsf{SSQ}(P_v \cup S) &= \sum_{w \in P_v \cup S - \{v\}} [d(v,w)]^2 + \sum_{\substack{u,w \in P_v \cup S - \{v\} \\ u \neq w}} [d(u,w)]^2 \\
&\leq Q_v + \sum_{\substack{u,w \in P_v \cup S - \{v\} \\ u \neq w}} 2\left([d(u,v)]^2 + [d(v,w)]^2\right) \quad \text{(by Lemma 5.7)} \\
&= Q_v + \sum_{w \in P_v \cup S - \{v\}} \sum_{u \in P_v \cup S - \{v,w\}} 2\,[d(v,w)]^2 \\
&= Q_v + 2(|S| + p - 2) \sum_{w \in P_v \cup S - \{v\}} [d(v,w)]^2 \\
&= Q_v + 2(|S| + p - 2)Q_v \\
&= (2(|S| + p) - 3)Q_v \\
&\leq (2(|S| + p) - 3)R_v \quad \text{(by Equation (19))} \\
&\leq 2\left[\frac{2(|S| + p) - 3}{|S| + p}\right] \mathsf{SSQ}(P^* \cup S) \quad \text{(by Equation (17))} \\
&= \left[4 - \frac{6}{|S| + p}\right] \mathsf{SSQ}(P^* \cup S).
\end{aligned}$$

This completes the proof. ■

6 Other Observations

6.1 Geometric Versions

In [AIKS91], Aggarwal et. al. present efficient algorithms for geometric versions of MDP and MVP problems. Their algorithms work for arbitrary dimensions. Here, we present an efficient algorithm for 1-dimensional MAP problem. We also point out some difficulties in extending the approach of [AIKS91] to obtain a polynomial time algorithm for the MAP problem for higher dimensions.

In the 1-dimensional version of MAP, the node set V consists of n points (denoted by x_1, x_2, ..., x_n) on the real line. Thus $d(x_i, x_j) = |x_i - x_j|$. We denote this problem by 1D-MAP. As mentioned earlier, minimizing the average distance is equivalent to minimizing the sum of the distances. So, we present our algorithm in terms of the latter objective. Our algorithm is based on the following definition and lemma.

Definition 6.1 Let $V = \{x_1, x_2, \ldots, x_n\}$ be a set of n points on the real line. A subset S of V is **contiguous** if and only if for every pair of points x and z in S with $x < z$, there does not exist a point $y \in V - S$ such that $x < y < z$.

Lemma 6.2 Given a 1D-MAP instance I consisting of a set V of n points and an integer p, every optimal solution P^* to I is a set of p contiguous points.

Proof: The proof is by contradiction. Let P^* be an optimal solution to I, and suppose P^* is not contiguous. Then by the above definition, there is a pair of points x and z in P^* and a point $y \in V - P^*$ such that $x < y < z$. Without loss of generality, assume that there is no point of P^* in the open interval (x, z); i.e., x is the largest value in P^* less than y and z is smallest value in P^* greater than y. It can be verified that one of x and z can be replaced by y to obtain a placement better than P^*. ■

Theorem 6.3 The 1D-MAP problem can be solved in $O(n \log n)$ time.

Proof: By Lemma 6.2, every optimal solution is a contiguous set of p points. Since there are only $n - p + 1$ such sets, it follows that an optimal solution can be found by considering each contiguous set with p points and choosing a set with the minimum value with respect to the SOD function.

To implement this algorithm in $O(n \log n)$ time, the points are first sorted into increasing order. Let $\{x_1, x_2, \ldots, x_n\}$ denote the points in increasing order. The $n - p + 1$ contiguous sets are then considered in order. We now indicate how the value of SOD function for a given set can be obtained in constant time, using the value of the preceding set.

Let $S_i = \{x_{i-p+1},\ x_{i-p+2},\ \ldots,\ x_i\}$ denote the contiguous set of p points ending at x_i $(p \le i \le n)$. By the definition of the SOD function,

$$\mathsf{SOD}(S_i) = \sum_{i-p+1 \le j < k \le i} d(x_j, x_k) \quad (p \le i \le n).$$

The goal is to find a set S_r for which $\mathsf{SOD}(S_r)$ is a minimum among the sets S_p, $S_{p+1}, \ldots, S_n$. To compute the values of $\mathsf{SOD}(S_i)$ $(p \le i \le n)$ efficiently, we define some auxiliary functions.

Let $\mathsf{SODL}(S_i)$ ($\mathsf{SODR}(S_i)$) denote the sum of the distances from each point in S_i to x_{i-p+1} (x_i), the leftmost (rightmost) point in S_i. In other words,

$$\mathsf{SODL}(S_i) = \sum_{j=i-p+2}^{i} d(x_{i-p+1}, x_j) \quad (p \le i \le n) \tag{21}$$

$$\mathsf{SODR}(S_i) = \sum_{j=i-p+1}^{i-1} d(x_j, x_i) \quad (p \le i \le n). \tag{22}$$

Using these equations, the following recurrences can be derived.

$$\begin{aligned}
\mathsf{SODL}(S_{i+1}) &= \mathsf{SODL}(S_i) - (p-1)d(x_{i-p+1}, x_{i-p+2}) + d(x_{i-p+2}, x_{i+1}) && (23)\\
\mathsf{SODR}(S_{i+1}) &= \mathsf{SODR}(S_i) - d(x_{i-p+1}, x_i) + (p-1)d(x_i, x_{i+1}) && (24)\\
\mathsf{SOD}(S_{i+1}) &= \mathsf{SOD}(S_i) - \mathsf{SODL}(S_i) + \mathsf{SODR}(S_{i+1}) && (25)
\end{aligned}$$

The value of $\mathsf{SOD}(S_p)$ can be computed in $O(p)$ time using the following equation.

$$\mathsf{SOD}(S_p) = \sum_{j=1}^{p-1} j(p-j)d(x_j, x_{j+1}) \tag{26}$$

The values of $\mathsf{SODL}(S_p)$ and $\mathsf{SODR}(S_p)$ can be computed in $O(p)$ time using Equations (21) and (22). Given the values of $\mathsf{SOD}(S_i)$, $\mathsf{SODL}(S_i)$ and $\mathsf{SODR}(S_i)$, Equations (23), (24) and (25) show how the values of $\mathsf{SOD}(S_{i+1})$, $\mathsf{SODL}(S_{i+1})$ and $\mathsf{SODR}(S_{i+1})$ can be computed in $O(1)$ time ($p \leq i < n$). Thus, a set S_r for which $\mathsf{SOD}(S_r)$ is a minimum among S_p, S_{p+1}, ..., S_n can be found in $O(n)$ time.

Therefore, the running time of the algorithm is dominated by the sorting step which takes $O(n \log n)$ time. ■

In the case of the 2-dimensional version of MAP (2D-MAP), it can be shown that every optimal solution P^* is such that every point inside the convex hull of P^* belongs to P^*, thereby extending the notion of contiguity to two dimensional space. Unfortunately, the number of such convex hulls can be exponential.

In [AIKS91], an optimal placement with respect to variance was found by computing the p^{th}-order Voronoi diagram, and finding a set with the smallest variance from among the sets corresponding to the Voronoi regions. This approach works because it was shown that there is a set P^* with minimum variance and a position z in the plane, such that P^* is the set of the p points nearest to z. However, the following example shows that there exists an instance of 2D-MAP which has the property that the set with the minimum sum of distances is not the set of p points nearest to *any* position in the plane.

Consider the sets $A = \{a_1, a_2, \ldots, a_{p/2}\}$ with each a_i placed at $(0,1)$, $B = \{b_1, b_2, \ldots, b_{p/2}\}$ with each b_i placed at $(0,-1)$, a point c placed at $(1-\epsilon, 0)$, a point d placed at $(-1+\epsilon, 0)$. Construct the instance of 2D-MAP with $V = A \cup B \cup \{c, d\}$, ϵ a small positive quantity, and the number of facilities to be placed being p. For $p \geq 4$, the optimal placement is $P^* = A \cup B$. However for each position z on the plane, the p points closest to z contain either c or d. If z is in the first quadrant, then the point c is closer to z than any point in B. A symmetric argument holds for the other quadrants. This example is valid, even when A and B are made sets of distinct geometric points clustered around $(0,1)$ and $(0,-1)$ respectively. Thus we observe that the sets corresponding to the Voronoi regions of a p^{th}-order Voronoi diagram do not correspond to an optimal solution. Thus, it does not seem possible to directly extend the approach of [AIKS91] for finding a placement with minimum variance to the problem of finding a placement that minimizes the sum of the distances.

6.2 Comparisons of Solutions Under the Three Objectives

In Section 5, we presented heuristics to obtain near-optimal solutions under the three objectives, namely minimum diameter, minimum average distance and minimum variance. This section is devoted to a comparison of the three objectives. We present problem instances (without node weights or distinguished nodes) to show that a solution which is optimal with respect to one objective can be arbitrarily poor with respect to another objective, even when the triangle inequality holds. These examples demonstrate that the objectives are in some sense orthogonal.

Our examples use suitable combinations of the following sets of nodes with distance functions. (The "combination" operation will be specified after the definitions of these sets.) The distance functions are symmetric and it is easy to verify that they satisfy the triangle inequality. In the following specifications, ϵ is an arbitrarily small positive quantity.

1. Let $X = \{x_1, x_2, \ldots, x_p\}$ be a set of nodes with $d(x_i, x_j) = 1$, $1 \le i < j \le p-1$.
2. Let $Y = \{y_0, y_1, y_2, \ldots, y_{p-1}\}$ with $d(y_0, y_i) = 1 + \epsilon$ for $1 \le i \le (p-1)$ and $d(y_i, y_j) = \epsilon$ for $1 \le i < j \le (p-1)$.
3. Let $Z = \{z_0, z_1, z_2, \ldots, z_{p-1}\}$ with $d(z_0, z_i) = (p-1)/2$ for $1 \le i \le (p-1)$ and $d(z_i, z_j) = \epsilon$ for $1 \le i < j \le (p-1)$.
4. Let $R = \{r_0, r_1, \ldots, r_{p-1}\}$ with $d(r_0, r_i) = \sqrt{p/2 - 1}$ for $1 \le i \le (p-1)$ and $d(r_i, r_j) = \epsilon$ for $1 \le i < j \le (p-1)$.
5. Let $W = \{w_0, w_1, w_2, \ldots, w_{p-1}\}$ with $d(w_0, w_i) = p/2 - 1$ for $1 \le i \le (p-1)$ and $d(w_i, w_j) = \epsilon$ for $1 \le i < j \le (p-1)$.
6. Let $T = \{t_0, t_1, t_2, \ldots, t_{p-1}\}$ with $d(t_0, t_i) = \sqrt{p/2 + 1}$ for $1 \le i \le (p-1)$ and $d(t_i, t_j) = \epsilon$ for $1 \le i < j \le (p-1)$.

For each set α defined above, Table 1 gives the values of $\mathsf{DIA}(\alpha)$, $\mathsf{SOD}(\alpha)$ and $\mathsf{SSQ}(\alpha)$. We will refer to this table several times in the following discussion.

For disjoint sets of nodes α and β with distance functions d_α and d_β respectively, define $\mathsf{COMBINE}(\alpha, \beta)$ as the graph with the node set $\alpha \cup \beta$ and the distance function $d_{\alpha\cup\beta}$ given by

$$\begin{aligned} d_{\alpha\cup\beta}(x, y) &= d_\alpha(x, y) && \text{if } x, y \in \alpha \\ &= d_\beta(x, y) && \text{if } x, y \in \beta \\ &= M && \text{if } x \in \alpha \text{ and } y \in \beta \end{aligned}$$

Set α	$\mathsf{DIA}(\alpha)$	$\mathsf{SOD}(\alpha)$	$\mathsf{SSQ}(\alpha)$
X	1	$p(p-1)/2$	$p(p-1)/2$
Y	$1+\epsilon$	$(p-1)((1+\epsilon)+(p-2)\epsilon/2)$	$(p-1)((1+\epsilon)^2+(p-2)\epsilon^2/2)$
Z	$(p-1)/2$	$(p-1)((p-1)/2+(p-2)\epsilon/2)$	$(p-1)((p-1)^2/4+(p-2)\epsilon^2/2)$
R	$\sqrt{p/2-1}$	$(p-1)(\sqrt{p/2-1}+(p-2)\epsilon/2)$	$(p-1)((p/2-1)+(p-2)\epsilon^2/2)$
W	$p/2-1$	$(p-1)((p/2-1)+(p-2)\epsilon/2)$	$(p-1)((p/2-1)^2+(p-2)\epsilon^2/2)$
T	$\sqrt{p/2+1}$	$(p-1)(\sqrt{p/2+1}+(p-2)\epsilon/2)$	$(p-1)((p/2+1)+(p-2)\epsilon^2/2)$

Table 1: Comparison of the Three Objectives

where M is a quantity which is much larger than any of the distances in α and β. It is easy to verify that if d_α and d_β satisfy the triangle inequality, then so does $d_{\alpha\cup\beta}$.

We use the sets and the COMBINE operation defined above to create instances that bring out the orthogonality of the measures. The reader should bear in mind that in each case, the number of facilities to be placed is p and that ϵ can be made arbitrarily small.

1. Diameter and Sum of the distances:
   Consider the graph COMBINE(X, Y). For this graph, the set X is the optimal placement with respect to diameter while the set Y is the optimal placement with respect to sum of the distances. From Table 1, it is easy to see that SOD(X) is arbitrarily worse than SOD(Y). Thus, for this graph, the set which is optimal with respect to diameter is arbitrarily poor with respect to sum of the distances.

2. Diameter and Sum of the squares of the distances:
   Here also the graph COMBINE(X, Y) suffices because X is optimal with respect to diameter and Y is optimal with respect to sum of the squares of the distances. From Table 1, it is seen that SSQ(X) is arbitrarily worse than SSQ(Y).

3. Sum of the distances and Diameter:
   For this case, we consider the graph COMBINE(X, Z). The set Z minimizes the sum of the distances while X minimizes the diameter. Observe from Table 1 that DIA(Z) is arbitrarily worse than DIA(X).

4. Sum of the squares of the distances and Diameter:

Consider the graph $\mathsf{COMBINE}(X, R)$. The set R minimizes the sum of the squares of the distances while X minimizes the diameter. Observe from Table 1 that $\mathsf{DIA}(R)$ is arbitrarily worse than $\mathsf{DIA}(X)$.

5. Sum of the distances and Sum of the squares of the distances:
   Consider the graph $\mathsf{COMBINE}(X, W)$. The set W minimizes the sum of the distances while X minimizes the sum of the squares of the distances. Observe from Table 1 that $\mathsf{SSQ}(W)$ is arbitrarily worse than $\mathsf{SSQ}(X)$.

6. Sum of the squares of the distances and Sum of the distances:
   Consider the graph $\mathsf{COMBINE}(X, T)$. The set X minimizes the sum of the squares of the distances while T minimizes the sum of the distances. Observe from Table 1 that $\mathsf{SOD}(X)$ is arbitrarily worse than $\mathsf{SOD}(T)$.

7 Conclusions

We considered the compact location problem under three different objectives and presented efficient approximation algorithms which provide a provably good performance guarantee when the distances satisfy the triangle inequality. Tables 2 and 3 summarize the current results on the complexity and approximability of the compact location problems considered in this paper.

The results presented here raise several open questions. The first question is whether the MAP problem can be solved in polynomial time in two and higher dimensions. Another open question is to find polynomial time approximation algorithms for MAP and MVP problems which provide better performance guarantees. It is also of interest to study the compact location problems using other functional combinations of edge and node weights.

Acknowledgements: We thank Dr. Gyan Bhanot (Thinking Machines Corporation, Cambridge, MA) for suggesting the processor allocation problem during a conversation. We also thank Professor Dorit Hochbaum (University of California, Berkeley) for helpful suggestions.

Problem	MDP	MAP	MVP
General	NP-hard (Prop. 3.1)	NP-hard (Prop. 3.1)	NP-hard (Prop. 3.1)
Triangle Inequality	NP-hard (Prop. 3.1)	NP-hard (Prop. 3.1)	NP-hard (Prop. 3.1)
1D Version	Efficiently solvable [AIKS91]	Efficiently solvable (Thm. 6.3)	Efficiently solvable [AIKS91]
2D Version	Efficiently solvable [AIKS91]	Open	Efficiently solvable [AIKS91]

Table 2: Complexity Results for Compact Location Problems

Problem	MDP		MAP		MVP	
	UB	LB	UB	LB	UB	LB
General	—	NGR (Prop. 3.2)	—	NGR (Prop. 3.2)	—	NGR (Prop. 3.2)
Triangle Inequality	2 (Thm. 5.2)	2 (Prop. 5.3)	2- $1/p$ (Thm. 5.5)	Open	4 - $6/p$ (Thm. 5.8)	Open

Table 3: Approximability Results for NP-hard Compact Location Problems

Notes: UB denotes the best known upper bound on the performance guarantee. LB denotes the lower bound on the performance guarantee; that is, the intrinsic limit assuming $\mathsf{P} \neq \mathsf{NP}$. NGR is an abbreviation for "No Guaranteed Ratio".

References

[AA+95] B. Awerbuch, Y. Azar, A. Blum and S. Vempala, "Improved Approximation Guarantees for Minimum-Weight k-Trees and Prize-Collecting Salesmen," *Proc. 27th Annual ACM Symposium on Theory of Computing* (STOC'95), Las Vegas, NV, May 1995, pp 277-283.

[AIKS91] A. Aggarwal, H. Imai, N. Katoh and S. Suri, "Finding k points with Minimum Diameter and Related Problems," *J. Algorithms*, Vol. 12, No. 1, March 1991, pp 38-56.

[AMO93] R. K. Ahuja, T. L. Magnanti and J. B. Orlin, *Network Flows: Theory, Algorithms and Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[An72] H. C. Andrews, *Introduction to Mathematical Techniques in Pattern Recognition*, Wiley-Interscience, New York, NY, 1972.

[BC+95] A. Blum, P. Chalasani and S. Vempala, "Constant Factor Approximation for Minimum-Weight k-Trees in the Plane," *Proc. of 27th Annual ACM Symposium on Theory of Computing* (STOC'95), Las Vegas, NV, May 1995, pp 294-302.

[BKP93] J. Bar-Ilan, G. Kortsarz and D. Peleg, "How to Allocate Network Centers", *J. Algorithms*, Vol. 15, No. 3, Nov. 1993, pp 385-415.

[BE95] M. Bern and D. Eppstein, "Approximation Algorithms for Geometric Problems," unpublished manuscript, January 1995.

[DF85] M. E. Dyer and A. M. Frieze, "A Simple Heuristic for the p-Center Problem," *Operations Research Letters*, Vol. 3, No. 6, Feb. 1985, pp 285-288.

[EN89] E. Erkut and S. Neuman, "Analytical Models for Locating Undesirable Facilities," *European J. Operations Research*, Vol. 40, 1989, pp 275-291.

[FG88] T. Feder and D. Greene, "Optimal Algorithms for Approximate Clustering", *Proc. 20th Annual ACM Symp. on Theory of Computing* (STOC'88), May 1988, pp 434-444.

[GH94] N. Garg and D. S. Hochbaum, "An $O(\log k)$ Approximation Algorithm for the k Minimum Spanning Tree Problem in the Plane," *Proc. 26th ACM Symp. on Theory of Computing* (STOC'94), Montreal, May 1994, pp 432-438.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co., San Francisco, CA, 1979.

[Go85] T. F. Gonzalez, "Clustering to Minimize the Maximum Intercluster Distance," *Theoretical Computer Science*, Vol. 38, 1985, pp 293-306.

[Ha75] J. A. Hartigan, *Clustering Algorithms*, Wiley, New York, NY, 1975.

[HM79] G. Y. Handler and P. B. Mirchandani, *Location on Networks: Theory and Algorithms*, MIT Press, Cambridge, MA, 1979.

[HS86] D. S. Hochbaum and D. B. Shmoys, "A Unified Approach to Approximation Algorithms for Bottleneck Problems," *J. ACM*, Vol. 33, No. 3, July 1986, pp 533-550.

[Ja92] J. JaJa, *An Introduction to Parallel Algorithms*, Addison-Wesley Publishing Co., New York, NY, 1992.

[KP93] G. Kortsarz and D. Peleg, "On Finding a Dense Subgraph," *Proc. 34th Annual Symp. on Foundations of Computer Science* (FOCS'93), Oct. 1993, pp 692-701.

[Le82] D. T. Lee, "On k-Nearest Neighbor Voronoi Diagrams in the Plane," *IEEE Trans. Comput.*, Vol. C-31, 1982, pp 478-487.

[LV92] J. -H. Lin and J. S. Vitter, "ϵ-Approximations with Minimum Packing Constraint Violation," *Proc. 24th Annual ACM Symp. on Theory of Computing* (STOC'92), May 1992, pp 771-782.

[MF90] P. B. Mirchandani and R. L. Francis, *Discrete Location Theory*, Wiley-Interscience, New York, NY, 1990.

[PS85] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, Inc., New York, NY, 1985.

[RRT94] S. S. Ravi, D. J. Rosenkrantz and G. K. Tayi, "Heuristic and Special Case Algorithms for Dispersion Problems," *Operations Research*, Vol. 42, No. 2, March-April 1994, pp 299-310. (An extended abstract of this paper appeared in *Proceedings of the 1991 Workshop on Algorithms and Data Structures* (WADS-91), Ottawa, Lecture Notes in CS, Vol. 519, Springer-Verlag, New York, NY, August 1991, pp 355-366.)

[RS+94] R. Ravi, R. Sundaram, M. V. Marathe, D. J. Rosenkrantz, and S. S. Ravi, "Spanning Trees Short or Small," *Proc. 5th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA'94), January 1994, pp 546-555. (Complete version to appear in *SIAM J. Discrete Mathematics.*)

[RSL77] D. J. Rosenkrantz, R. E. Stearns and P. M. Lewis II, "An Analysis of Several Heuristics for the Traveling Salesman Problem," *SIAM J. Computing*, Vol. 6, No. 3, 1977, pp 563-581.

[Ta91] A. Tamir, "Obnoxious Facility Location on Graphs," *SIAM J. Discrete Mathematics*, Vol. 4, 1991, pp 550-567.