Compact Location Problems *

(Extended Abstract)

V. Radhakrishnan *S.O. Krumke*† *M.V. Marathe* *D.J. Rosenkrantz* *S. S. Ravi*

Department of Computer Science
University at Albany – SUNY
Albany, NY 12222

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 a processor allocation problem in multiprocessor systems.

We consider the problem under three different objectives, namely minimizing the diameter, minimizing the average distance, and minimizing the variance. In general, the problem is NP-hard under any of the objectives. We observe that in general, even obtaining a relative approximation for any of the objectives is NP-hard.

We present a general framework for approximating the minimum cost compact location problem for each of the above measures. We present efficient approximation algorithms for the three objectives when the distances satisfy the triangle inequality. For minimizing the diameter, our heuristic provides a performance guarantee of 2. We also show that no polynomial time heuristic for minimizing the diameter can provide a better performance guarantee unless P=NP. For minimizing the average distance and variance, our heuristics provide performance guarantees of $(2 - 2/p)$ and $(4 - 6/p)$ respectively. Our framework can be extended to the case when there are both node weights and edge weights, where our approximation algorithms yield the same performance guarantee as in the edge weighted case. Our algorithms can be further generalized when we are given node weights, edge weights and a set of distinguished nodes, where we are required to place a facility at each distinguished site, and must place p additional facilities so as to minimize some cost measure.

We also show that the bounds given for these heuristics are tight. Our algorithms are easy to parallelize. We also present polynomial time algorithms when the given network is a tree.

1 Introduction

We consider several problems dealing with the placement of a specified number of facilities on the vertices of a given network so as to minimize some function of the distances between the facilities. When the vertices correspond to points in Euclidean space, these location problems model several situations arising in statistical clustering [Ha75] and pattern recognition [An72]. This type of location problem is also relevant to the following processor allocation problem in multiprocessor systems. Consider a computational task consisting of a number of communicating subtasks. At a given time, some of the processors may be already allocated and the remaining processors are available. The problem is to select a subset of processors from the currently available processors, one per subtask, such that the cost of communication among the processors executing the subtasks is minimized[1].

*Research Supported by NSF Grants CCR-89-03319 and CCR-90-06396.

†Current Address: Universität Würzburg, Am Hubland, W-8700, Würzburg, Germany.

[1] We thank Dr. Gyan Bhanot (Thinking Machines Corporation, Cambridge, MA) for suggesting the problem in a conversation.

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 and communication costs among the offices, etc.

The goal in the above mentioned location problems is usually to minimize some notion of "cost" associated with the placement. The cost may reflect the price of locating a facility at a certain point (captured as node weights) or the cost of transportation between facilities (sum of edge weights) or the maximum time it requires to communicate between any two facilities (maximum edge weight or bottleneck cost). Often, finding a placement which minimizes one or more cost measures is NP-hard.

Given the apparent intractability of such placement problems in general networks, we investigate the existence of approximation algorithms with good performance guarantee. We present a general framework for finding a near optimal placement of facilities on a given network, so as to minimize one or more of the above cost measures. The framework can be used to handle both edge weights and node weights simultaneously. Furthermore, we can also handle the case when there is a set of distinguished sites which already have facilities placed and we wish to place the remaining facilities so as to minimize the total cost of placement.

We also show that when the given network is a tree, optimal solutions to several problems can be found in polynomial time.

1.1 Minimum Cost Compact Location

Consider the following location problem: Given an undirected graph $G = (V, E)$ with n vertices, and with a non-negative distance between every pair of vertices, 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 optimality criterion 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 possible criterion 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 facilities). By a simple reduction from the clique problem we can show that

Proposition 1.1 *The problems MDP, MAP and MVP are NP-hard, even when the distances satisfy the triangle inequality.* $\square$

We first make a simple observation about approximation algorithms for the general case. Recall that a relative approximation algorithm always produces a solution within a constant factor of the optimum.

Proposition 1.2 *If the distances are not required to satisfy the triangle inequality, then there are no polynomial time relative approximation algorithm for the problems MDP, MAP and MVP unless* $\mathbf{P} = \mathbf{NP}$.

The above negative result motivates the quest for an efficient relative approximation algorithm for the MDP problem when the distances do satisfy the triangle inequality. As the following Theorem shows, we can obtain an approximation algorithm with a performance guarantee of 2 for the MDP problem. Further, no polynomial time algorithm can provide a better performance guarantee unless $\mathbf{P} = \mathbf{NP}$.

Theorem 1.3 *Given a undirected complete graph with edge weights satisfying the triangle inequality, there is a polynomial time algorithm $Heur-MDP$ which produces a placement for the MDP problem with a performance guarantee of 2. Moreover, unless $\mathbf{P} = \mathbf{NP}$, no polynomial time algorithm can provide a placement with a diameter of no more than $\alpha \cdot OPT(I) + \beta$ for any $\alpha < 2$ and $\beta > 0$. Here $OPT(I)$ denotes the diameter of an optimal solution for instance I.*

Our method is powerful enough to generalize to the case when we want to find a minimum cost placement of facilities with two other cost measures, namely the average distance and variance of the placement.

Theorem 1.4 *Given an undirected complete graph with edge weights satisfying triangle inequality, there are polynomial time algorithms $Heur - MAP$ and $Heur - MVP$ which produce placements for the problems MAP and MVP with performance guarantees $2(1 - 1/p)$ and $(4 - 6/p)$ respectively. Here p is the number of facilities to be placed.*

1.2 Extension to the node-weighted case

We can strengthen each of the results above by considering nonnegative costs on the nodes and aiming to minimize the sum of diameter and the maximum node weight or sum of average distance and the average node weight or sum of the variance and the sum of node weights. We denote these problems by MDP_{wt}, MAP_{wt} and MVP_{wt} respectively. The case when only node weights are present is not very interesting as one can easily find polynomial time solutions to the problems by simply sorting the nodes in increasing order of their weights. Hence we aim at finding placement of facilities so as to optimize a function of both the node and edge weights. Extending our techniques used to prove the performance guarantee for the edge weighted case we prove that:

Theorem 1.5 *Given an undirected complete graph on n nodes, with nonnegative node and edge weights satisfying the triangle inequality, there are polynomial time algorithms $Heur-MDP_{wt}$, $Heur-MAP_{wt}$ and $Heur-MVP_{wt}$ which produce placements for the problems MDP_{wt}, MAP_{wt} and MVP_{wt} with performance guarantees $2-\frac{\omega^*}{\omega^*+d^*}$, $2(1 - 1/p)$ 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 the optimal placement.*

1.3 Extension to the distinguished nodes case

All the above results can be further extended to the case when we are given a complete graph with node weights and edge weights satisfying triangle inequality and also a *distinguished set* of nodes $S \subseteq V$. The task then is to place $p + |S|$ facilities such that there is one facility at each node in S. The remaining p facilities can be placed on any of the remaining nodes. At first it may appear that the problem would be harder than the unconstrainted case. By modifying our technique used to prove the previous results we show that the distinguished facilities case can be approximated with the same performance guarantee.

Theorem 1.6 *Given an undirected complete graph on n nodes with nonnegative node weights and edge weights satisfying the triangle inequality, and a set of distinguished nodes $S \subseteq V$, with $|V - S| \geq p$, there are polynomial time algorithms $Heur{-}MDP^D_{wt}$, $Heur{-}MAP^D_{wt}$ and $Heur{-}MVP^D_{wt}$ which produce a placement for the problems MDP^D_{wt}, MAP^D_{wt} and MVP^D_{wt} with a performance guarantee of $2 - \frac{\omega^*}{\omega^* + d^*}$, $2(1-1/(p+|S|))$ and $(4-6/(p+|S|))$ 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 the optimal placement.*

1.4 Special Cases

Given that the compact location problems are hard for arbitrary networks, we investigate the complexity of these problems when the underlying topology is more restricted. We consider the case when the underlying graph is a tree. We show that the problems MAP and MDP have a polynomial time algorithm. In this case the distances between two nodes is the length of the shortest path between them. Formally we show that,

Theorem 1.7 *Given an undirected tree with edge weights, there are polynomial time algorithms Tree-MDP and Tree-MAP which produce optimal placements for problems MDP and MAP respectively.*

We can extend the above theorem to the case with node weights and distinguished nodes. Thus:

Theorem 1.8 *Given an undirected tree with edge weights, there are polynomial time algorithms $Tree - MDP^D_{wt}$ and $Tree - MAP^D_{wt}$ which produce optimal placements for problems MDP^D_{wt} and MAP^D_{wt} respectively.*

The remainder of this paper is organized as follows. Section 2 contains some definitions and preliminaries. Section 3 contains a discussion of related work. Section 4 addresses the edge weighted case and the node weighted case. Section 5 presents our results for the compact location problems with distinguished nodes. Section 6 discusses our results when the underlying graph is restricted to be a tree. Due to space limitations, only sketches of proofs are included in this extended abstract. Detailed proofs will appear in a complete version of this paper.

2 Preliminaries

As mentioned earlier, a network is specified by a set $V = \{v_1, v_2, \ldots, v_n\}$ of nodes and a non-negative and symmetric distance $d(v_i, v_j)$ between every pair of distinct nodes. 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 DIA(P), is given by $\text{DIA}(P) = \max_{\substack{\{x,y\}, x \neq y \\ x,y \in P}} d(x,y)$. The **sum of the distances** in P, denoted by SOD(P), is given by $\text{SOD}(P) = \sum_{\substack{\{x,y\}, x \neq y \\ x,y \in P}} d(x,y)$. The **sum of the squares of the distances** in P, denoted by SSQ(P), is given by $\text{SSQ}(P) = \sum_{\substack{\{x,y\}, x \neq y \\ x,y \in P}} [d(x,y)]^2$.

We note that the average distance and the variance [2] [AIKS91] of a placement P are equal to $\frac{2}{p(p-1)}$ SOD(P) and $\frac{2}{p(p-1)}$ SSQ(P) respectively. Therefore, obtaining a placement P with minimum average distance is

[2] In [AIKS91], the fraction is $\frac{1}{p}$. We use $\frac{2}{p(p-1)}$ for reasons of uniformity.

equivalent to obtaining a placement with the minimum value of SOD(P); similarly, obtaining a placement P with minimum variance is equivalent to obtaining a placement with the minimum value of SSQ(P). Furthermore, for two placements P and Q, the ratios SOD(P)/SOD(Q) and SSQ(P)/SSQ(Q) are equal to the corresponding ratios for average distance and variance, respectively. For these reasons, we present our results for the minimum average distance and minimum variance criteria using the functions SOD and SSQ defined above.

Formal specifications of the problems considered in this paper are given below.

(1) Problems with Edge Weights

(a) Minimum Diameter Placement (MDP)

Instance: A set $V = \{v_1, v_2, \ldots, v_n\}$ of n nodes, a non-negative 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 DIA(P) is minimized.

(b) Minimum Average Distance Placement (MAP)

Instance: As in MDP above.

Requirement: Find a subset P of V, with $|P| = p$, such that 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 SSQ(P) is minimized.

(2) Edge and Node Weights

(a) Minimum Diameter Placement with Node Weights (MDP_{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 nonnegative distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, and an integer p.

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

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

is minimized.

(b) Minimum Average Distance Placement with node weights (MAP_{wt})

Instance: As in MDP_{wt} above.

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

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

is minimized.

(c) Minimum Variance Placement with node weights (MVP_{wt})

Instance: As in MDP_{wt} above.

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

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

is minimized.

(3) Edge Weights and Distinguished Nodes

(a) Minimum Diameter Placement with Distinguished Nodes (MDP^D)

Instance: A set $V = \{v_1, \ldots, v_n\}$ of n nodes with a nonnegative distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, a subset $S \subset V$ 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 $\mathrm{DIA}(P \cup S) = DIA(P \cup S)$ is minimized.

(b) Minimum Average Distance Placement with distinguished nodes (MAP^D)

Instance: As in MDP^D above.

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

(c) Minimum Variance Distance Placement with distinguished nodes (MVP^D)

Instance: As in MDP^D above.

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

(4) Edge weights, Node Weights and Distinguished Nodes

(a) Minimum Diameter Placement with Node weights and Distinguished Nodes (MDP^D_{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 nonnegative distance $d(v_i, v_j)$ for each pair v_i, v_j of nodes, a subset $S \subset V$ 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

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

is minimized.

(b) Minimum Average Distance Placement with node weights and distinguished nodes (MAP^D_{wt})

Instance: As in MDP^D_{wt} above.

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

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

is minimized.

(c) Minimum Variance Distance Placement with node weights and distinguished nodes (MVP^D_{wt})

Instance: As in MDP^D_{wt} above.

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

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

is minimized.

The distances specified in instances of the above problems are said to satisfy the **triangle inequality** if for any three distinct nodes v_i, v_j and v_k, $d(v_i, v_j) + d(v_j, v_k) \geq d(v_i, v_k)$.

3 Related Work

In contrast to the NP-hardness results contained in Section 1, 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, the 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. Similarly, the clustering problems considered in the literature [HoS86, FG88, Go85] involve partitioning the given set of nodes into clusters so as to minimize a given objective function. The location problems considered in this paper are of a different flavor since the objective functions involve only a subset of nodes. Facility location problems where the objective is to place facilities so as to *maximize* some function of the distances between facilities have been considered in the literature; see for example [EN89, RRT91]. Problems in which the placement of facilities is not restricted to the nodes of the network have also been studied [MF90]. We consider only location problems in which the facilities are placed at the nodes of the network.

We also consider algorithms which minimize some function of node weights and edge weights. Hochbaum and Shmoys [HoS86] and Dyer and Frieze [DF85] consider the node weighted versions of center problem. Location problems with other optimizing criteria have also been considered in the literature. Lin and Vitter [LV92] provide approximations for the s-median problem where s median nodes must be chosen so as to minimize the sum of the distances from each node to its nearest median. The solution method 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 and Peleg [BP91] consider the balanced center problem. They provide approximation algorithms for problem of allocating network centers wherein each center is allowed to service only a bounded number of nodes.

4 Approximation Results

In this section we discuss our results for the compact location problems when we have both edge weights and node weights. Throughout this section we assume that the edge satisfy the triangle inequality. All our approximations make use of a generic procedure. This generic algorithm is given in Figure 1. As can be seen the algorithm has three parameters: the graph instance $G(V, E)$ with node and edge weights, the set S of distinguished nodes, and F, a function mapping subsets of V of cardinality p into $\mathbb{R}^+$, depending on the problem instance.

To illustrate our ideas, we first sketch our proof for the node weighted minimum diameter placement problem.

Due to space limitations, we omit the analyses for the node weighted minimum average placement and node weighted minimum variance problems. We then sketch the proof for node weighted minimum variance problem with distinguished nodes.

4.1 Approximating the diameter plus the maximum node weight

Here, we consider the MDP_{wt} problem. Recall that in this case we are given a complete graph on n nodes with node weights and edge weights satisfying triangle inequality. We are required to place p facilities so as to minimize the sum of maximum edge weight and the maximum node weight. The approximation algorithm $Heur - MDP_{wt}$ is shown in Figure 2. The algorithm is reminiscent of the generic procedure given for other bottleneck problems by Hochbaum and Shmoys [HoS86]. It returns a certificate of failure if a feasible solution is not found. We now define the concept of a bottleneck subgraph used in the algorithm.

Definition 4.1 *Let $V = \{v_1, \cdots, v_n\}$ be a set of nodes, let $d(v_i, v_j)$ be a nonnegative distance on $V \times V$ and let $\omega(v)$ be a nonnegative node-weight function on V. For $c \in \mathbb{R}^+$, we define* **the bottleneck subgraph**, *$BOTT(V, c)$ to be the subgraph induced by the set of nodes $v \in V$ such that $\omega(v) \leq c$. The distances in $BOTT(V, c)$ and the node-weights coincide with those in V.*

Theorem 4.2 *Let I be an instance of MDP_{wt}. Let $P^* \subseteq V$ be an optimal placement and let $P \subset V$ be the placement produced by $Heur - MDP_{wt}$. Then $DIA_{wt}(P)/DIA_{wt}(P^*) \leq \left(2 - \frac{\omega^*}{d^*+\omega^*}\right)$, where d^* is the maximum distance between two nodes in P^* and $\omega^* = \max\{\omega(v) : v \in P^*\}$ is the maximum node–weight in P^*.*

Proof: We look at the iteration where $Heur - MDP_{wt}$ considers BOTT(V, ω^*). By the definition of ω^*, it follows that all the nodes of P^* are contained in BOTT(V, ω^*). Consequently $Gen - Alg_{wt}$ can find a solution and will not return a "certificate of failure".

Let $v \in P^*$ be an arbitrary node and consider the iteration of $Gen - Alg\_wt^D$, where v is looked at. Let $P_v = \{w_1, \cdots, w_{p-1}\}$ be the set of nearest neighbors of v in $W - \{v\}$. By the definition of BOTT(V, ω^*), 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

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

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

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

PROCEDURE $Gen - Alg^{D}_{wt}(G(V,E),S,F)$
/* $G(V,E)$ denotes the graph with edge and/or node weights. S is the distinguished set of nodes. F is the measure with respect to which we choose the placement of nodes. */

1. **If** $|V - S| < p$ **Then Return** "certificate of failure" produced
2. **Else**
   Begin
   (a) Solution $\leftarrow \emptyset$; Value $\leftarrow +\infty$
   (b) **For** each vertex $v \in V$ **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 the $p-1$ vertices 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$\leftarrow P_v$
         Value$\leftarrow t$

      End

   End
3. **Output** Solution and Value

Figure 1: Generic Algorithm

PROCEDURE $Heur-MDP_{wt}$

1. Solution$\leftarrow \emptyset$; Value $\leftarrow +\infty$
2. Sort the node–weights in increasing order. Let the result be $\{\omega_1, \ldots, \omega_n\}$.
3. **For** $i := 1$ **To** n **Do**

   Begin

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

   (b) Run $Gen-Alg^D_{wt}$ on W with $D = \phi$ and $F(X) = \mathrm{DIA}(X)$. Let the result be P.

   (c) **IF** P is not a "certificate of failure" **then** $t \leftarrow \mathrm{DIA}_{\mathrm{wt}}(P)$
   else $t \leftarrow +\infty$

   (d) **If** $t <$Value **then** Value$\leftarrow t$
   Solution$\leftarrow P$

   End

4. **Output** Solution and Value

Figure 2: Procedure for the problem MDP_{wt}

Since $Heur-MDP_{wt}$ picked a set P which was minimum among all possible sets, it now follows that $DIA_{wt}(P) \leq DIA_{wt}(P_v) \leq 2\left(1 - \frac{\omega^*}{d^*+\omega^*}\right) \cdot DIA_{wt}(P^*)$. $\Box$

We remark that the heuristic $Heur-MDP_{wt}$ can be implemented in parallel.

Proposition 4.3 *$Heur-MDP_{wt}$ can be implemented to run in $O(\log n)$ parallel time using $O(\max\{n^3, n^2p^2/\log p\})$ processors.*

4.2 Approximating variance plus sum of node weights in the presence of special nodes

Here, we consider the problem MVP^D_{wt}. Recall that this problem requires minimizing the sum of variance and the node weights for a placement in which one is required to place facilities at the distinguished nodes.

The analysis of this case requires an extended form of triangle inequality which we state below.

Lemma 4.4 *If the distances in a network satisfy the triangle inequality, then for all vertices x, y, z*

$$[d(x,z)]^2 \leq 2\left([d(x,y)]^2 + [d(y,z)]^2\right).$$

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

$$[d(x,z)]^2 \leq [d(x,y)]^2 + [d(y,z)]^2 + 2d(x,y)d(y,z). \tag{5}$$

Moreover, we obtain from

$$0 \leq (d(x,y) - d(y,z))^2 = [d(x,y)]^2 + [d(y,z)]^2 - 2d(x,y)d(y,z) \tag{6}$$

that

$$2d(x,y)d(y,z) \leq [d(x,y)]^2 + [d(y,z)]^2 . \tag{7}$$

Thus by (5) and (7)

$$[d(x,z)]^2 \leq [d(x,y)]^2 + [d(y,z)]^2 + 2d(x,y)d(y,z) \leq 2\left([d(x,y)]^2 + [d(y,z)]^2\right), \tag{8}$$

which proves the lemma □

Next we give a transformation of the problem MVP^D_{wt} into the problem MVP^D. This transformation has the property that a solution to the transformed MVP^D instance with performance guarantee α implies a solution with the same performance guarantee α for the MVP^D_{wt} problem. Given, this our task then is to prove the required bound for the edge weighted case.

Theorem 4.5 *Given an instance I of MVP^D_{wt}, we can construct an instance I' of MVP^D in polynomial time with the following property: Given a placement for the instance I' of the MVP^D problem with performance guarantee α, the same placement provides a solution to the instance I of the MVP^D_{wt} problem with the same performance guarantee α.*

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

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

First we will show that the triangle inequality is not violated in I'. Let $\lambda = (p+|S|)/2$ then for arbitrary nodes $u, v, w \in V'$

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

We now prove that for a feasible placement $P \in V - S$ we have value $SSQ(P \cup S) = SSQ_{wt}(P \cup S)$, where $SSQ(P \cup S)$ uses distance distance function d' and $SSQ_{wt}(P \cup S)$ uses distance distance function d.

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

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

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

We consider the three sums in the last equality.

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

The second sum is treated as follows

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

Finally, the third sum equals

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

Putting together these results yields

$$\begin{aligned}SSQ(P\cup S) &= \frac{2}{(p+|S|)(p+|S|-1)}\left(X+\sum_{v\in P,s\in S}[d(v,s)]^2+\sum_{v,w\in P}[d(v,w)]^2+\right. \qquad (9)\\ &\quad \left.+1/2(p+|S|)(p+|S|-1)\left(Y+\sum_{v\in P}\omega(v)\right)\right)\\ &= SSQ_{wt}(P\cup S).\end{aligned}$$

Thus, feasible placements for I are exactly the feasible placements for I'. It follows that P^* is optimal for I if and only if P^* is optimal for I', and 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'. □

The algorithm again is the same generic procedure outline in Figure 1. We now give a theorem which gives the required performance guarantee for MVP_{wt}. By Theorem 4.5, this performance guarantee also applies to MVP^D_{wt}.

Theorem 4.6 *Let I be an instance of MVP^D. Let $P^*\subset V-S$ be an optimal placement and let $P\subset V-S$ be the placement produced by $Gen-Alg^D_{wt}$ with $F(X)=SSQ(X\cup S)$. Then $SSQ^D(P\cup S)/SSQ^D(P^*\cup S)\leq 4-6/(p+|S|)$.*

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

$$R_w = \sum_{v \in P^* \cup S} [d(v,w)]^2 . \tag{10}$$

We have

$$SSQ^D(P^* \cup S) = \frac{1}{2} \sum_{w \in P^* \cup S} R_w . \tag{11}$$

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

$$R_v = \min_{w \in P^* \cup S} R_w . \tag{12}$$

Then

$$SSQ^D(P^* \cup S) = \frac{1}{2} \sum_{w \in P^* \cup S} R_w \geq \frac{1}{2} \sum_{w \in P^* \cup S} R_v = \frac{p + |S|}{2} R_v . \tag{13}$$

This yields

$$R_v \leq \frac{2}{p + |S|} \mathrm{SSQ}^{\mathrm{D}}(P^* \cup S). \tag{14}$$

Consider now the iteration in the loop of the algorithm, where Step 2(b) considers node v. Let $P_v = \{w_1, \ldots, w_k\}$ be the set of nearest neighbors of v in $V - S - \{v\}$, where $k = p$ if $v \in S$ and $k = p - 1$ if $v \in P^*$. Define

$$Q_v = \sum_{w \in P_v \cup S - \{v\}} [d(v,w)]^2 . \tag{15}$$

By definition of R_v and Q_v we have

$$Q_v = \sum_{s \in S} [d(v,s)]^2 + \sum_{w \in P_v} [d(v,w)]^2 \leq \sum_{s \in S} [d(v,s)]^2 + \sum_{w \in P^*} [d(v,w)]^2 = R_v , \tag{16}$$

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

We prove the theorem by showing that

$$SSQ^D(P_v \cup S) \leq (4 - 6/(p + |S|)) SSQ^D(P^* \cup S). \tag{17}$$

The theorem then follows. The estimation of $SSQ^D(P_v \cup S)$ is done as follows:

$$\begin{aligned}
SSQ^D(P_v \cup S) &= \sum_{w \in P_v \cup S - \{v\}} [d(v,w)]^2 + \sum_{\substack{u,w \in P_v \cup S - \{v\} \\ u \neq w}} [d(u,w)]^2 \\
&\leq Q_v + \sum_{\substack{u,w \in P_v \cup S - \{v\} \\ u \neq w}} 2 \left([d(u,v)]^2 + [d(v,w)]^2 \right) \\
&= Q_v + \sum_{w \in P_v \cup S - \{v\}} \sum_{u \in P_v \cup S - \{v,w\}} 2 \, [d(v,w)]^2 \\
&= Q_v + 2(|S| + p - 2) \sum_{w \in P_v \cup S - \{v\}} [d(v,w)]^2 \\
&= Q_v + 2(|S| + p - 2) Q_v \\
&= (2(|S| + p) - 3) Q_v \overset{(16)}{\leq} (2(|S| + p) - 3) R_v \\
&\leq 2 \frac{2(|S| + p) - 3}{|S| + p} \mathrm{SSQ}^{\mathrm{D}}(P^* \cup S) \\
&= (4 - \frac{6}{|S| + p}) SSQ^D(P^* \cup S).
\end{aligned}$$

This completes the proof. □

5 Location Problems Restricted to Trees

Here we consider compact location problems where the input is restricted to be a tree structure. In this case the distances between two vertices correspond to path lengths along the trees.

Definition 5.1 *A tree based distance structure τ is a set $V = \{v_1, v_2, \cdots v_n\}$ of n vertices, a spanning tree T on these vertices, and a non-negative length assigned to each edge of the tree. For each pair v_i, v_j of vertices, the distance $d(v_i, v_j)$ implied by τ is the sum of the edge lengths along the unique path in T connecting v_i and v_j.*

Versions of compact location problems can be defined for trees, in the same manner as we defined for arbitrary graphs but the distances are now specified by a tree-based distance structure. We denote these problems by Tree-MDP, Tree-MAP and Tree-MVP respectively. For instance, for the Tree-MDP problem the input is a tree based distance structure and an integer p. The requirement is to find a subset consisting of p nodes, whose diameter is minimized.

We now outline a polynomial time algorithm to solve the Tree-MAP problem optimally. Our algorithm uses dynamic programming to compute such a placement.

Definition 5.2 *Given a tree structure τ, a set of nodes A is* **closed** *if and only if for each $x, y \in A$, every node in the (unique) tree path between x and y in in A.*

Next, we state two lemmas which show that among all the optimal solutions to the $Tree - MDP$ and $Tree - MAP$ problem, there exist solutions which are closed. We prove a more general version of the Lemma 5.3 in the Appendix.

Lemma 5.3 *Among the optimal solutions to the Tree-MAP is one which is a closed set of nodes.*

Lemma 5.4 *Among the optimal solutions to the Tree-MDP is one which is a closed set of nodes.*

Theorem 5.5 *Tree-MAP can be solved optimally in $O(p^2 n)$ time.*

Proof: We use a dynamic programming based procedure to obtain our solution. Envision the tree as being rooted at an arbitrarily selected node, say node r. The algorithm processes the tree in a bottom up manner. First the leaf nodes are processed. Then, a non-leaf node is processed after all its children have been processed.

Consider a set A of i nodes, such that $1 \leq i \leq p$, where A is contained in the subtree rooted at some node z. Define

$$COST(A) = \sum_{\{x,y\}, x \neq y, x,y \in A} d(x,y) + (p-i) \sum_{x \in A} d(x,z)$$

Consider a set B of $(p-i)$ nodes, where B contains no nodes in the subtree rooted at node z. Define

$$COST(B) = \sum_{\{x,y\}, x \neq y, x,y \in B} d(x,y) + i \sum_{x \in B} d(x,z)$$

Let $P = A \cup B$. Then **SOD(P)**$= COST(A) + COST(B)$. When the bottom up algorithm processes node z, it finds the minimum cost possible for the $COST(A)$ under the requirement that A is closed and A contains z. It will do this for each value of i. If the subtree headed by node z contains at least i nodes, then define $COST_i^z = \min_A \{COST(A) | A$ is a closed set of i nodes in the subtree headed by z and $z \in A\}$.

If the subtree headed by z contains fewer than i nodes, then define $COST_i^z = \infty$.

Now we consider the following computation of values. Suppose node z is a leaf. Then

$$COST_1^z = 0$$

$$COST_i^z = \infty \;\; 2 \leq i \leq p$$

Now suppose that z is a non-leaf. Let $\delta(z)$ be the number of children of z, and let $c_1, \cdots, c_{\delta(z)}$ denote these children, listed in some arbitrary order. Define $COST_i^{z,j}$ for $1 \leq i \leq p$, $1 \leq j \leq \delta(z)$ as follows:
$COST_i^{z,j} = \min_A \{COST(A) | A$ is a compact set of $i, z \in A$, and every other node in A lies in one of the subtrees rooted at $c_1, \cdots c_j\}$.

The values of $COST_i^{z,j}$ can now be computed as follows:

$$COST_1^{z,1} = 0$$

$$COST_{i+1}^{z,1} = i(p-i)w(z, c_1) + COST_i^{c_1} \;\; 2 \leq i \leq p$$

$$COST_i^{j,j+1} = \min\left(COST_i^{z,j}, \min_{t<i}\left(COST_{i,t}^{z,j} + COST_t^{z,j} + COST_t^{c_j+1} + t(p-t)d(z, c_j)\right)\right)$$

for $1 \leq i \leq p, 2 \leq j \leq \delta(z)$. Finally, for the node z, we do the following

$$COST_i^z = COST_i^{z,\delta(z)}, 1 \leq i \leq p$$

The optimal solution to the Tree-MAP problem instance has a cost given by the minimal value of the $COST_p^z$, over all nodes z. It can be seen that we can find an optimal solution by the above method in $O(p^2 n)$ time. □

We omit the proofs for the problems $Tree-MDP_{wt}^D$ and $Tree-MAP_{wt}^D$ due to the lack of space. The proofs for the problems is more involved. We can define the concept of $S-close$ which is similar to the concept of closeness defined for the problems MAP and MDP. A brief sketch of the proof for the problem MAP^D appears in the Appendix.

References

[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.

[AHU] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading MA., 1974.

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

[BP91] J. Bar-Ilan and D. Peleg, “Approximation algorithms for selecting network centers (Preliminary version),” *Proc. 2nd Workshop on Algorithms and Data Structures* (WADS ’91) (Springer-Verlag Lecture Notes in CS, Vol. 519), Ottawa, Canada, Aug. 1991, pp 343-354.

[CLR90] T.H Cormen, C.E. Leiserson and R.L. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 1990.

[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. of Operations Research*, Vol. 40, 1989, pp 275-291.

[FG88] T. Feder, D. Greene, “Optimal Algorithms for Approximate Clustering”, *ACM Symposium on Theory of Computing*, 1988, pp 434-444.

[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.

[HoS86] D. S. Hochbaum and D. B. Shmoys, “A Unified Approach to Approximation Algorithms for Bottleneck Problems,” *JACM*, 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.

[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 STOC* (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.

[RRT91] S. S. Ravi, D. J. Rosenkrantz and G. K. Tayi, “Heuristic and Special Case Algorithms for Dispersion Problems,” *Proc. 2nd Workshop on Algorithms and Data Structures* (WADS ’91) (Springer-Verlag Lecture Notes in CS, Vol. 519), Ottawa, Canada, Aug. 1991, pp 355-366. (Complete version to appear in *Operations Research*.)

6 Appendix

We briefly discuss the problem $Tree - MAP^D$. We first define the concept of $S - openness$ for a placement of facilities and then prove that among the optimal solution for the problem $Tree - MAP^D$ there is a solution which is $S - closed$. Using this we can use a similar dynamic programming technique as described for the problem without distinguished nodes to show that one can obtain an optimal solution to the problem $Tree - MAP^D$.

Definition 6.1 *Given a tree based distance structure τ with node set V, and a set of distinguished nodes $S \subseteq V$, a set of nodes $A \subseteq V - S$ is $S - closed$ if $\forall x, y \in A$, every node in the (unique) tree path between x and y is in $A \cup S$.*

Lemma 6.2 *Among the optimal solutions to an instance T of the Tree$-MAP^D$ with distinguished node set S, there is one which is an $S - closed$ set of nodes.*

Proof Consider a set of $P \subseteq V - S$. Define the $Openness^S$ of a pair of nodes x and y in P to be the number of nodes along the (unique) path between x and y that are not in $P \cup S$. Let

$$Openness^S(P) = \sum_{\{x,y\} \in P} Openness^S(x, y)$$

Among the optimal solutions to the $Tree - MAP^D(I)$, let P be one of minimal $Openness^S$. Assume that $Openness^S(P)$ is non-zero. We will show that this leads to a contradiction.

Let $q = p + |S|$. Note that deletion of any edge from tree τ partitions the tree into two disconnected subtrees. Since $Openness^S(P)$ is nonzero, there is an edge (u, v) in the tree τ such that $u \in P \cup S$, and $v \notin P \cup S$, and if edge (u, v) were to be deleted from τ, the subtrees containing u and v would each contain at least one member of P, and the subtree containing u would contain a number j of nodes from $P \cup S$, such that $j \leq q/2$. Without loss of generality, we can assume that the subtree containing u contains a node $x \in P$ such that every node along the path between x and u is a member of $P \cup S$, and that x does not occur as an intermediate node along any path between two members y and z of $P - \{x\}$. Thus node x is a leaf of the subtree of τ consisting of members of P and the intermediate nodes in the paths connecting members of P. Let $P' = P - \{x\} \cup \{v\}$. Then

$$SOD(P') = SOD(P) - \sum_{w \in P \cup S - \{x\}} d(w, x) + \sum_{w \in P \cup S - \{x\}} d(w, v)$$

Let A be the members of $P \cup S - \{x\}$ that are in the subtree containing u, and let B be the members in the subtree containing B. Then $|A| = j - 1$, and $|B| = q - j$. For each node $w \in A$, $d(w, v) \leq d(w, x) + d(x, v)$. So $d(w, v) - d(w, x) \leq d(x, v)$. For each node $w \in B$ $d(w, x) \leq d(w, v) + d(x, v)$. So $d(w, x) - d(w, v) \leq d(x, v)$. So

$$SOD(P') = SOD(P) - \sum_{w \in B} d(w, x) - d(w, v) \sum_{w \in A} d(w, v) - d(w, x)$$

$$\leq SOD(P) - (q - j)d(x, v) + (j - 1)d(x, v)$$

$$= SOD(P) - (q - 2j + 1)d(x, v)$$

Since $j \leq q/2$, it follows that $q - 2j + 1 > 0$, so $SOD(P') \leq SOD(P)$. If $SOD(P') < SOD(P)$, there is an immediate contradiction to the assumption that P is an optimal solution to the $Tree\_MAP^D(I)$. So assume $SOD(P') = SOD(P)$, in which case P' is an optimal solution. Consider $Openness^S(P')$. Becasue of the manner in which X was selected, for each $y, z \in P - \{x,\}$, $Openness^S(y, z)$ is no greater for P' than for P. For each node $w \in A \cap P$, $Openness^S(w, v)$ for P' is the same as $Openness^S(w, x)$ for P. For each node $w \in B \cap P$, $Openness^S(w, x)$ for P' is less than $Openness^S(w, x)$ for P. Since $B \cap P$ is non-null, $Openness^S(P') < Openness^S(P)$, contradicting the assumption that P is a minimal $Openness^S$ solution. $\square$