SOLVING NONLINEAR SINGLE-FACILITY NETWORK LOCATION PROBLEMS

JOHN HOOKER
Carnegie-Mellon University, Pittsburgh, Pennsylvania

(Received December 1984, revisions received May, October 1985, January 1986; accepted February 1986)

We present a general approach to solving nonlinear single-facility network location problems. We assume that cost is any convex function of distances, and solve this class of problem by solving convex subproblems on "treelike segments" into which the network is decomposed. We show that only a fraction of treelike segments generally need be examined, and that our method results in a reasonably efficient general-purpose algorithm. The algorithm is particularly effective on real-world (as opposed to random) networks, and when the cost function is nondecreasing and "semiseparable," as are many popular cost functions. Finally, we describe theoretical complexity bounds and computational experience.

Network location models have the advantage that their reticular structure fits a large number of applications and is relatively easy to conceptualize. But they have hitherto had the disadvantage that solution algorithms have been devised only for problems with linear or a few specific nonlinear cost functions. Furthermore, each type of cost function has required a different algorithm. No unified approach to the solution of nonlinear network location problems has been proposed.

Our purpose is to present such a unified approach to the solution of *single-facility* network location problems. We will describe a reasonably efficient all-purpose algorithm that solves single-facility problems in which cost is any convex function of distances. This category includes all problems for which algorithms already exist. The algorithm, unlike its predecessors, also accommodates a wide variety of side constraints. Its execution time is a low order polynomial function of problem size, and computational experience suggests it can comfortably solve problems on real-world networks of two or three hundred arcs.

Although the algorithm assumes a convex cost function, this is not a serious limitation. Levy (1967) and Handler and Mirchandani (1979) have shown that, if cost is a concave and additively separable function of distances, there must be an optimal location at a node. Thus single-facility concave problems do not need a solution algorithm in any nontrivial sense, because one can solve them simply by examining the nodes. It appears that the great majority of cost functions likely to arise in practice are either convex or else concave and additively separable. If so, our algorithm should be adequate for the great majority of applications that require a nontrivial algorithm.

There are potentially a large variety of location problems that admit nonlinear models. Two important nonlinear problems—minimization of nonlinear transport costs and maximization of accessibility—generally call for concave, additively separable cost functions and thus do not require our algorithm. But we have described elsewhere (1984, 1985) a wide variety of convex, nonlinear cost functions that are potentially useful and for which a solution algorithm now exists. Many are appropriate to public-sector contexts, since equity-sensitive cost functions tend to be convex and nonlinear. Location problems for obnoxious facilities (e.g., toxic waste dumps, pollution control facilities, nuclear power plants), which arise in both public and private sector settings, also generally have convex cost functions. The well-known 1-center, 1-median cent-dian problems have convex cost functions. Our algorithm is also useful for accommodating complicated side constraints in problems that were previously soluble only without them.

Our general approach is to exploit the fact that location problems are easier to solve on trees than on general networks. This ease of solution traces mainly to the convexity (in a sense to be defined) of the distance function on trees. Fortunately, there is a simple way to decompose any network into "treelike segments" on which the distance function is likewise convex, and in fact linear. We solve location problems on general networks by solving convex subproblems on these treelike segments. Since the number of segments grows with the cube of the number of nodes (in the worst case), we show how to avoid solving subproblems on all but a fraction of the segments. The result is a reasonably efficient all-purpose algorithm.

We begin in Section 1 with a statement of the

Subject classification: 185, 481 location on networks

0030-364X/86/3405-0732 $01.25

problem and a review of its history. Section 2 develops the notion of a treelike segment. Section 3 shows how to solve location problems by solving convex subproblems on treelike segments. The next two sections describe how this basic algorithm can be improved: Section 4 indicates how to eliminate entire arcs, without solving subproblems on them, with an upper bounding technique based on the convexity of the cost function; and Section 5 proves the surprising fact that if each convex subproblem is solved on a shortest-path tree rooted at a treelike segment, rather than just on the segment itself, several other treelike segments can be eliminated in the process. This result is based on the premise that the cost function is nondecreasing and "semiseparable" as well as convex. It and the upper-bounding technique are incorporated in an improved algorithm. Section 6 describes the large class of popular cost functions that are nondecreasing and semiseparable. Section 7 indicates how the algorithm can be modified slightly to accommodate side constraints. Section 8 determines worst-case complexity of the algorithm, and Section 9 reports computational experience with a FORTRAN implementation.

1. The Problem and Its History

Let G be the set of points on a network, and let $V = \{v_1, \ldots, v_n\}$ be the set of nodes. Consider the arcs (v_i, v_j) to be rectifiable, and for any $x, y \in G$, let $d(x, y)$ be the distance from x to y along any shortest path from x to y. We wish to solve the problem

$$\min z(x) = f(d(x, v_1), \ldots, d(x, v_n)), \quad x \in D \subseteq G, \qquad (1)$$

with *objective function* z: $D \rightarrow R$ and convex *cost function* f: $R^n \rightarrow R$.

The unconstrained case of problem (1) (i.e., $D = G$) has been solved for certain cost functions f. Hakimi (1964, 1965) solved the *1-median* problem, for which f is a linear function $f(d_1, \ldots, d_n) = \sum_{i=1}^n w_i d_i$ with each $w_i \geq 0$, as well as the *1-center* problem, for which f is the weighted maximum function $f(d_1, \ldots, d_n) = \max_i\{w_i d_i\}$ and each $w_i \geq 0$. Goldman and Dearing (1975), Church and Garfinkel (1978) and Minieka (1983) solved obnoxious-facility 1-median and 1-center problems, for which each $w_i \leq 0$. Halpern (1976, 1978, 1980) solved the cent-dian problem, in which f is a convex combination of the 1-median and 1-center cost functions. Shier and Dearing (1983) provided necessary and sufficient conditions for local optima when the cost function is an L_p norm of distances ($1 \leq p \leq \infty$). Most of their results can be extended to other cost functions, but they describe no method for enumerating local optima so as to find a global optimum.

Single-facility (as well as p-facility) problems are generally easier to solve on trees. Goldman (1971) efficiently solved the 1-median problem, Handler (1973) the unweighted 1-center problem, and Dearing and Francis (1974) the weighted 1-center problem (with addends) on trees. Also, Dearing (1977), Francis (1977), and Tansel et al. (1982) have solved on trees the nonlinear 1-center problem, whose cost function has the form $f(d_1, \ldots, d_n) = \max_i\{g_i(d_i)\}$.

The relative ease of solving location problems on trees traces largely to the fact that the distance function $d(\cdot, x)$: $G \rightarrow R$ is convex on G for all $x \in G$ if and only if G is a tree. Dearing, Francis and Lowe (1976) proved this result. For any $x, y \in G$ they define $L_\alpha(x, y)$ to be the set of $w \in G$ that satisfy $d(x, y) = d(x, w) + d(w, y)$ and $d(x, w) = \alpha d(x, y)$. The *line segment* $L(x, y)$ from x to y is the union of $L_\alpha(x, y)$ over all $\alpha \in [0, 1]$. A set $D \subseteq G$ is *convex* if $L(x, y) \subseteq D$ for all $x, y \in D$. A function z: $D \rightarrow R$ is *convex* on a convex set D if $z(w) \leq (1 - \alpha)z(x) + \alpha z(y)$ for all $w \in L_\alpha(x, y)$, all $\alpha \in [0, 1]$, and all $y \in D$.

Theorem 1. (Dearing, Francis and Lowe). *The function $d(\cdot, x)$: $G \rightarrow R$ is convex for all $x \in G$ if and only if G is a tree.*

This result is useful for the following reason. Let f: $D \subseteq R^n \rightarrow R$ be *nondecreasing* if $d_i' \geq d_i$ for all i implies $f(d_1', \ldots, d_n') \geq f(d_1, \ldots, d_n)$ for all $(d_1, \ldots, d_n)$, $(d_1', \ldots, d_n') \in R^n$. It is easy to show that:

Lemma 1. *If $y_1, \ldots, y_n \in G$ and $d(\cdot, x)$ is convex on G for each $x \in G$, then for any convex, nondecreasing f: $R^n \rightarrow R$, the function z: $G \rightarrow R$ given by $z(x) = f(d(x, y_1), \ldots, d(x, y_n))$ is convex.*

Also,

Lemma 2. (Dearing, Francis and Lowe). *If z: $D \rightarrow R$ is convex on $D \subseteq G$, a local minimum of z is a global minimum.*

Thus if G is a tree, D is convex, and the cost function f is convex and nondecreasing, then any locally optimal solution of (1) is globally optimal. A local optimum can readily be found, if by no other method, simply by "homing in" on it—by moving x along G in a direction that diminishes $z(x)$ until no such direction can be found (Handler and Mirchandani, p. 180).

2. Treelike Segments

The key to solving location problems on general networks is to decompose the networks into "treelike segments." We first define the notion of a "treelike set" and later use it to define a treelike segment, which is essentially its closure.

We say that points x, y on an arc "belong to the same treelike set" if the shortest paths to any node from x are the same as those from y, except for any portion of the paths lying between x and y. In Figure 1, for instance, x_1 and x_2 belong to the same treelike set, but x_2, x_3 and x_4 belong to three distinct treelike sets. It is clear that in a tree, any two points interior to an arc lie in the same treelike set, whence the name "treelike set."

These ideas can be made precise as follows. Given arc $A = (u, w)$ connecting nodes u and w, and given points $x, y \in A$, let $A(x, y)$ denote the portion of A strictly between x and y, and let $A[x, y]$ denote the closure of $A(x, y)$. Let $b(x, y)$ be the length of $A[x, y]$, with $b = b(u, w)$. Then $x, y \in A(u, w)$ belong to the *same treelike set* if for every $v_i \in V$,

$$L(x, v_i) - A[x, y] = L(y, v_i) - A[x, y]. \qquad (2)$$

It is evident from Figure 1 that if we define $d_i(x) = b(u, x) + d(u, v_i)$ and $d_i'(x) = b - b(u, x) + d(w, v_i)$, then

$$d(x, v_i) = \min\{d_1(x), d_i'(x)\}, \qquad (3)$$

and similarly for y. If $d_i(x) < d_i'(x)$, the shortest paths from x to v_i all contain node u; if $d_i'(x) < d_i(x)$, they contain w; if $d_i(x) = d_i'(x)$, they contain both u and w. If we set $s_i = \frac{1}{2}[d(w, v_i) - d(u, v_i) + b]$, the first case is equivalent to $b(u, x) < s_i$, the second case to $b(u, x) > s_i$, and the third case to $b(u, x) = s_i$. It is straightforward to show the following lemma.

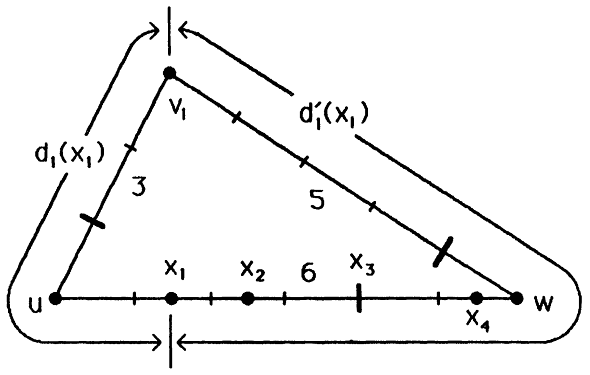

Figure 1. Illustration of treelike sets.

Lemma 3. *Given arc A from u to w, let the nodes of a network be numbered so that $s_1, \ldots, s_n$ is a nondecreasing sequence. Then points $x, y \in A(u, w)$ belong to the same treelike set if and only if one of the following holds:*

$$s_j = b(u, x) = b(u, y) \quad \textit{for some} \quad j \in \{1, \ldots, n\}, \qquad (4)$$

(in which case $x = y$), or

$$s_j < b(u, x) < s_{j+1} \quad \textit{and} \quad s_j < b(u, y) < s_{j+1}, \quad \textit{for some} \quad j \in \{1, \ldots, n-1\}. \qquad (5)$$

Proof. Suppose first that (4) or (5) holds. Since (4) implies $x = y$, the lemma is trivially true. Thus it remains to show that (5) implies (2) for all $v_i \in V$. If $i \leq j$, we have from (5) and the previous remarks that all shortest paths from both x and y to v_i contain w, so that (2) follows. A similar argument applies for $i > j$.

We now suppose that (2) holds for all i and wish to show that either (4) or (5) holds. Since $0 < b(u, x) < b$ and s_i ranges from zero (e.g., when $v_i = w$) to b (when $v_i = u$), the following cases are exhaustive.

Case I. $b(u, x)$ is s_i for some $i \in \{1, \ldots, n\}$. Here, u, $w \in L(x, v_i)$, so that by (2) u, $w \in L(y, v_i)$ as well. Thus $b(u, y) = s_i$ and (4) follows.

Case II. $s_i < b(u, x) < s_{i+1}$ for some $i \in \{1, \ldots, n-1\}$. Here, all shortest paths from x to v_i contain w, and all from x to v_{i+1} contain u. The former statement and (2) imply that all shortest paths from y to v_i contain w, so that $s_i < b(u, y)$. By similar reasoning, the latter statement implies $b(u, y) < s_{i+1}$, and (5) follows.

The relation of belonging to the same treelike set is obviously reflexive and symmetric; Lemma 3 establishes its transitivity. It is thus an equivalence relation and partitions the interior of any arc into *treelike sets.* The following result is corollary of Lemma 3.

Corollary 1. *Let $s_0' \ldots, s_{m+1}'$ be the distinct s_i's in Lemma 3, in nondecreasing order, where $0 \leq m \leq n - 2$. Let $z_0, \ldots, z_{m+1}$ be the points of $A(u, w)$ satisfying $b(u, z_i) = s_i'$ for $i = 0, \ldots, m + 1$. Then the treelike sets in $A(u, w)$ are the singletons $\{z_i\}$ for $i = 1, \ldots, m$ and the open intervals $A(z_i, z_{i+1})$ for $i = 0, \ldots, m$.*

We now define a *treelike segment* to be the closure of any open treelike set. Figure 1, for instance, contains six treelike segments, bounded by the nodes and the three heavy tick marks. In general, we have from

Corollary 1:

Corollary 2. *The treelike segments on an arc $A = A[u, w]$ consist of at least one and at most $n - 1$ adjacent closed intervals jointly covering A, with boundaries at $z_0, \ldots, z_{m+1}$ (as defined in Corollary 1).*

3. Basic Solution Algorithm

We can now solve the unconstrained version of (1) by minimizing $z(x)$ on each treelike segment S and selecting the smallest minimum. That is, we solve (6) for each treelike segment $S = A[x_1, x_2]$.

$$\min_{x \in S} z(x) = f(d(x, v_1), \ldots, d(x, v_n)). \qquad (6)$$

To see why these subproblems are relatively easy to solve, we need the following lemma, which is easily proved.

Lemma 4. *Let g: $R \to R^n$ be given by $g(t) = (g_1(t), \ldots, g_n(t)$, where each g_i is linear, and let f: $R^n \to R$ be convex. Then h: $R \to R$, given by $h(t) = f(g(t))$, is convex.*

We next define h: $[0, b] \to R$ by

$$h(d(u, y)) = f(d(y, v_1), \ldots, d(y, v_n)) \qquad (7)$$

for any $y \in A$. Solving (6) is clearly equivalent to solving the line search problem,

$$\text{minimize}\{h(t)\}$$

$$\text{subject to } b(u, x_1) \leqslant t \leqslant b(u, x_2). \qquad (8)$$

Problem (8) in turn is relatively easy to solve because of the following lemma.

Lemma 5. *Function h in (8) is convex on $[b(u, x_1), b(u, x_2)]$.*

Proof. Suppose again that the nodes are ordered so that $s_1, \ldots, s_n$ is a nondecreasing sequence. Then, by Lemma 3 and (3), we have for some $j \in \{1, \ldots, n\}$,

$$\begin{aligned} &f(d(y, v_1), \ldots, d(y, v_n)) \\ &\quad = f(d(u, v_1) + t, \ldots, d(u, v_j) + t, \\ &\qquad\qquad d(w, v_{j+1}) + b - t, \ldots, d(w, v_n) + b - t). \qquad (9) \end{aligned}$$

provided $y \in A[x_1, x_2]$. Thus the arguments of f in (7) are linear functions of t. From this result and the convexity of f we infer by Lemma 4 that h is convex on $[b(u, x_1), b(u, x_2)]$.

We now present an algorithm for solving (1). It incorporates a simple technique that permits one to exclude the facility from certain parts of the network. One can exclude the facility from any arc by considering that arc as "infeasible" and passing over it in Step 1 of the algorithm. Also, one can exclude the facility from an open interval within an arc by inserting a node at either end of the interval and regarding the new arc between the nodes as infeasible. We postpone discussion of more complicated side constraints to Section 7.

Step 0. Set the treelike segment counter k to zero.

Step 1. Choose any unexamined feasible arc $A = A[u, w]$ of G; if none remain, go to Step 3. Compute $d(u, v_i)$ and $d(w, v_i)$ for $i = 1, \ldots, n$, and use the results to compute $s_1, \ldots, s_n$. Sort these in nondecreasing order and determine $s'_0, \ldots, s'_{m+1}$ so as to locate the treelike segments on $A[u, w]$ as described in Corollary 2.

Step 2. For each treelike segment $A[x_1, x_2]$ on A, let $k = k + 1$ and find a solution t^* of the convex line search problem (8); denote the corresponding point on A by x_k^*. When all the segments of A are enumerated, go to Step 1.

Step 3. Any x_j^* such that $z(x_j^*) = \min_{1 \leqslant l \leqslant k}\{z(x_l^*)\}$ solves (1).

4. Improvement with an Upper Bounding Technique

It is generally possible to improve substantially the solution method of the previous section by using an upper bounding technique. The idea is simply to eliminate any arc on which we know that the objective function $z(x)$ cannot achieve a value lower than a previously established upper bound on its optimal value. We show in this section how to proceed when f is differentiable or is the weighted maximum function. The method must be adapted to other nondifferentiable functions on a case-by-case basis.

When a treelike segment is examined, we obtain a locally optimal point x_k^*, as in Section 3. Let $z_{\min}$ be the minimum value of $z(x_k^*)$ over all treelike segments k examined so far. Thus $z_{\min}$ is an upper bound on the global minimum of $z(x)$. When a new arc $A = [u, w]$ is considered, we attempt to show as follows that $z(x) \geqslant z_{\min}$ for all $x \in A$, so that A may be eliminated.

First, let $d_i = d(u, v_i)$ and $d'_i = d(w, v_i)$ for $i = 1, \ldots, n$. When f is differentiable, we let $f_i(d_1, \ldots, d_n)$ be $\partial f/\partial d_i$ evaluated at $(d_1, \ldots, d_n)$ and appeal to the following result.

Lemma 6. *Let f in (1) be differentiable, and let $x \in A = A[u, w]$, with $t = b(u, x)$. If $v_1, \ldots, v_m$ are the nodes*

v_i satisfying $s_i < b = b(u, w)$ on arc A, then

$$z(x) \geq z(u) - ta, \tag{10}$$

where

$$a = \sum_{i=1}^{m} |f_i(d_1 \ldots, d_n)| - \sum_{i=m+1}^{n} f_i(d_1, \ldots, d_n). \tag{11}$$

Proof. Note that for any $x \in A$, there is a shortest path from v_i, $i = m + 1, \ldots, n$, to x that contains u. Thus $d(x, v_i) = d_i + t$ for $i = m + 1, \ldots, n$. Also $d(x, v_i) = d_i - t_i$ for some t_i, $i = 1, \ldots, m$, where $-t \leq t_i \leq t$. Thus

$$\begin{aligned} z(x) &= f(d(x, v_1), \ldots, d(x, v_n)) \\ &= f(d_1 - t_1, \ldots, d_m - t_m, d_{m+1} + t, \ldots, d_n + t). \end{aligned} \tag{12}$$

But since f is convex and differentiable, we have for any vector d, $f(d + \Delta d) \geq f(d) + \nabla f(d) \cdot \Delta d$. Thus

$$\begin{aligned} &f(d_1 - t_1, \ldots, d_m - t_m, d_{m+1} + t, \ldots, d_n + t) \\ &\quad \geq f(d_1, \ldots, d_n) - \sum_{i=1}^{m} t_i f_i(d_1, \ldots, d_n) \\ &\qquad + t \sum_{i=m+1}^{n} f_i(d_1, \ldots, d_n) \\ &\quad \geq f(d_1, \ldots, d_n) - t \sum_{i=1}^{m} |f_i(d_1, \ldots, d_n)| \\ &\qquad + t \sum_{i=m+1}^{n} f_i(d_1, \ldots, d_n). \end{aligned}$$

The last inequality follows from the fact that $-t \leq t_i \leq t$. This result and (12) imply (10).

When f is the weighted maximum function, $f(d_1, \ldots, d_n) = \max_i\{w_i d_i\}$ and each $w_i \geq 0$, we appeal to an analogous result.

Lemma 7. *Let f in* (1) *be the weighted maximum function, and let x, A, t and $v_1, \ldots, v_m$ be as in Lemma* 6. *Then* (10) *holds when*

$$a = \begin{cases} w_j & \text{if } 1 \leq j \leq m, \\ -w_j & \text{if } m + 1 \leq j \leq n, \end{cases} \tag{13}$$

where $w_j d_j = \max_{1 \leq i \leq n}\{w_i d_i\}$.

Proof. It is clear that

$$\max\left\{\max_{1 \leq i \leq m}[w_i(d_i - t)], \max_{m+1 \leq i \leq n}[w_i(d_i + t)]\right\}$$

$$\leq \begin{cases} w_j d_j - t w_j & \text{if } 1 \leq j \leq m, \\ w_j d_j + t w_j & \text{if } m + 1 \leq j \leq n. \end{cases} \tag{14}$$

Also,

$$z(x) = \max_{1 \leq i \leq m}\{w_i d(x, v_i)\}$$

$$\leq \max\left\{\max_{1 \leq i \leq m}[w_i(d_i - t)], \max_{m+1 \leq i \leq n}[w_i(d_i + t)]\right\}.$$

This result and (14) imply (10).

The following theorem provides a lower bound z_0 for $z(x)$ on A. If $z_0 \geq z_{\min}$, we eliminate A from consideration.

Theorem 2. *Let f in* (1) *either be differentiable or be the weighted maximum function. Let x lie on arc $A = A[u, w]$, with $t = b(u, x)$, and let $v_1, \ldots, v_m$ be the nodes v_i satisfying $s_i < b$ on arc A. Let a be given by* (11) *or* (13), *and let a' be given by the same formula, with d_i' replacing d_i for $i = 1, \ldots, n$. Then*

$$z(x) \geq z_0$$

$$= \begin{cases} z(u) - [z(u) - z(w) + a'b]a/(a + a') & \\ \qquad\qquad \text{if } a, a' > 0 & (15) \\ \min\{z(u), z(w)\} \qquad \text{otherwise.} & (16) \end{cases}$$

Proof. By Lemmas 6 and 7, we have that $z(x) \geq z(u) - ta$ and $z(x) \geq z(w) - (b - t)a'$. These lower bounds describe two lines above which $z(x)$ must lie, as in Figure 2, where $h(t)$ is as defined by (6). If a, $a' > 0$, then $z(x)$ is everywhere bounded below by z_0 if the lines intersect at (t_0, z_0). Setting $z(u) - t_0 a = z(w) - (b - t_0)a'$, we get $t_0 = [z(u) - z(w) + ba']/(a + a')$. Since $z_0 = z(u) - t_0 a$, (15) immediately follows. If $a < 0$ or $a' < 0$, then (16) obviously holds.

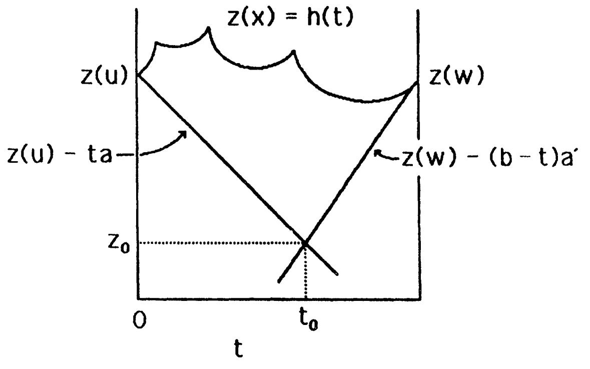

Figure 2. A lower bound for $z(x)$ on an arc.

5. A Further Improvement for Nondecreasing, Semiseparable Problems

We now describe a second improvement of the basic algorithm. It applies whenever the cost function f is nondecreasing and "semiseparable" as well as convex.

Semiseparability is a weakening of additive separability and is defined as follows. For any permutation π of n elements, let $f_\pi(x_1, \ldots, x_n) = f(x_{\pi(1)}, \ldots, x_{\pi(n)})$. Let 1_n and 0_n be vectors of n ones and n zeros, respectively. Then define $f: D \subseteq R^n \to R$ to be *semiseparable* if for all permutations π of n elements,

$$f_\pi(x + \Delta x, y + 1_t, z)$$
$$\geqslant f_\pi(x + \alpha 1_s + \Delta x, y, z) \quad (17)$$

implies $f_\pi(x, y + \alpha 1_t, z) \geqslant f_\pi(x + \alpha 1_s, y, z)$,

for all $x, \Delta x \in R^s$, all $y \in R^t$, all $z \in R^u$, and all integers $s, t, u \geqslant 0$ with $s + t + u = n$, where $\alpha \geqslant 0$ and $\Delta x \geqslant 0_s$, and all the arguments of f_π remain in D. In Section 6 we will describe the large class of useful cost functions that are semiseparable.

If S is a treelike segment on network G, let $T(S)$ be any *shortest-path tree rooted at S.* That is, $T(S)$ is a spanning tree of G containing S and for any $x \in S$, the path in $T(S)$ from x to any $v_i \in V$ is a shortest path in G from x to v_i. Given $T = T(S)$, let d^T be the distance function on $T(S)$.

Our strategy is to solve each convex subproblem on $T(S)$ rather than just on S, since this approach may allow us to eliminate some treelike segments. That is, instead of solving (6), we solve

$$\min_{x \in T(S)} z^T(x) = f(d^T(x, v_1), \ldots, d^T(x, v_n)). \quad (18)$$

Since f is nondecreasing, Lemma 1 applies that z^T is indeed convex. The key to elimination lies in the following theorem.

Theorem 3. *Let z be defined as in* (1), *with f nondecreasing and semiseparable. Let S be a treelike segment of network G, and let x_s solve* (18). *Then z(y) is nondecreasing as y moves along the path of T(S) from x_s to and including S.*

Proof. It suffices to show that z is nondecreasing on every treelike segment on the path of $T(S)$ from x_s to and including S. Let $A[y_1, y_2]$ be any interval contained in any such segment, with y_1 closer to S in $T(S)$ and y_2 closer to x_s. It suffices to show that $z(y_1) \geqslant z(y_2)$ (Figure 3).

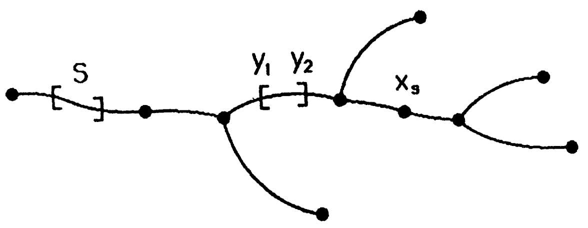

Figure 3. Illustration of Theorem 3.

We first write

$$z^T(y_2) = f(d^T(y_2, v_1), \ldots, d^T(y_2, v_n))$$
$$= f(d_1^T, \ldots, d_n^T), \quad (19)$$

setting $d_i^T = d^T(y_2, v_i)$ for brevity. Cut $T(S)$ at y_2 and let $v_i, \ldots, v_k$ be the nodes in the part of $T(S)$ containing S (excluding y_2 if y_2 is a node). Then we have

$$z^T(y_1)$$
$$= f(d_1^T - \Delta, \ldots, d_k^T - \Delta, d_{k+1}^T$$
$$+ \Delta, \ldots, d_n^T + \Delta), \quad (20)$$

where Δ is the length of $A[y_1, y_2]$. Note that $d^T(y_1, v_i) = d(y_1, v_i)$ and $d^T(y_2, v_i) = d(y_2, v_i)$ for $i = k + 1, \ldots, n$. Thus we can write

$$z(y_2) = f(d(y_2, v_1), \ldots, d(y_2, v_n))$$
$$= f(d_1, \ldots, d_k, d_{k+1}^T, \ldots, d_n^T), \quad (21)$$

where $d_i = d(y_2, v_i)$.

Now, since f is nondecreasing, z is convex on $T(S)$. Thus since x_s minimizes z^T on $T(S)$, we have

$$z^T(y_1) \geqslant z^T(y_2). \quad (22)$$

This result, (19) and (20) imply,

$$f(d_1^T - \Delta, \ldots, d_k^T - \Delta, d_{k+1}^T + \Delta, \ldots, d_n^T + \Delta)$$
$$\geqslant f(d_1^T, \ldots, d_n^T). \quad (23)$$

But since $d_i \leqslant d_i^T$ for all i, and f is semiseparable, (23) implies

$$f(d_1 - \Delta, \ldots, d_k - \Delta, d_{k+1}^T + \Delta, \ldots, d_n^T + \Delta)$$
$$\geqslant f(d_1, \ldots, d_k, d_{k+1}^T, \ldots, d_n^T). \quad (24)$$

Since y_1 and y_2 lie in a single treelike segment, it is clear that for $i = 1, \ldots, k$, $d(y_1, v_i) = d(y_2, v_i) \pm \Delta$, so that $d(y_1, v_i) = d_i - \Delta_i$ for some $\Delta_i \leqslant \Delta$, and therefore

$$z(y_1) = f(d_1 - \Delta_1, \ldots, d_k - \Delta_k,$$
$$d_{k+1}^T + \Delta, \ldots, d_n^T + \Delta). \quad (25)$$

But since f is nondecreasing, we have from (24) that

$$f(d_1 - \Delta_1, \ldots, d_k - \Delta_k, d_{k+1}^T + \Delta, \ldots, d_n^T + \Delta)$$
$$\geq f(d_1, \ldots, d_k, d_{k+1}^T, \ldots, d_n^T).$$

This result, (21) and (25) imply that $z(y_1) \geq z(y_2)$, as desired.

The utility of Theorem 3 is that finding the solution x_s of the location problem on $T(S)$ immediately eliminates all treelike segments that lie wholly on the path in $T(S)$ from x_s to and including S, for no point in any of these segments can be superior to x_s. Furthermore, finding x_s is a relatively easy task because, as remarked in Section 1, the single-facility problem is easily solved on a tree when the cost function is convex and nondecreasing, as it is here. Thus one can solve (1) by selecting uneliminated treelike segments S and solving the single-facility problem on $T(S)$; if the solution x_s is in S, it is recorded as the kth candidate x_k^* for a globally optimal location on G, and S is eliminated; if x_s is elsewhere, S and all the segments between S and x_s are eliminated. Any location x_j^* satisfying $z(x_j^*) = \min_{1 \leq l \leq k}\{z(x_l^*)\}$ is a global optimum.

The following algorithm incorporates th basic algorithm (Section 3) and the two improvements (Sections 4 and 5), using either where applicable. For convenience we say that *Technique 1* is the upper bounding technique of Section 4 and *Technique 2* is the elimination technique described in this section.

The algorithm is written so as to solve (18) in two stages: it first identifies on which arc $A_l = A[\hat{u}, \hat{w}]$ the solution x_s lies and then locates x_s on A_l. It does the latter by solving a line search problem similar to (8), namely,

$$\text{minimize } \{h^T(t)\},$$
$$\text{subject to } 0 \leq t \leq b(\hat{u}, \hat{w}) = \hat{b}. \tag{26}$$

In this formulation, h^T is defined by

$$h^T(d^T(\hat{u}, y)) = f(d^T(y, v_1), \ldots, d^T(y, v_n)). \tag{27}$$

We can cut $T(S)$ at the center of A_l and let $v_1, \ldots, v_j$ be the nodes in the half of $T(S)$ containing $\hat{u}$. Then we have an analogue of (9),

$$f(d^T(y, v_1), \ldots, d^T(y, v_n))$$
$$= f(d^T(\hat{u}, v_1) + t, \ldots, d^T(\hat{u}, v_j) + t,$$
$$d^T(\hat{w}, v_{j+1}) + \hat{b} - t, \ldots, d^T(\hat{w}, v_n) + \hat{b} - t), \tag{28}$$

so that the arguments of f in (27) are linear functions of t. Thus we can argue as in the proof of Lemma 5 that h^T is convex on $[0, \hat{b}]$.

Like the basic algorithm of Section 3, the algorithm below permits one to exclude the facility from certain arcs or intervals within arcs. In the latter case, a node is inserted at either end of each excluded interval, so as to create an arc. Each excluded arc is regarded as "infeasible" and passed over in Step 1 of the algorithm.

Step 0. Set the relative minimum counter k to zero and let $z_{\min} = \infty$.

Step 1. Choose any uneliminated feasible arc $A_j = A[u, w]$ of G; if none remain, go to Step 11. Compute $d_i = d(u, v_i)$ and $d_i' = d(w, v_i)$ for $i = 1, \ldots, n$, and record the location in G of the corresponding shortest-path trees, which we may call $T(u)$ and $T(w)$, respectively. If a list L_j of the treelike segments of A_j has already been made, go to Step 4. Otherwise compute $s_i = \frac{1}{2}[d_i' - d_i + b(u, w)]$ for $i = 1, \ldots, n$. If Technique 1 is not used, go to Step 3.

Step 2. Compute z_0 for A_j as described in Theorem 2. If $z_0 \geq z_{\min}$, eliminate A_j and go to Step 1.

Step 3. Sort $s_1, \ldots, s_n$ in nondecreasing order so as to determine the boundaries $s_0' \ldots, s_{m+1}'$ of A_j's treelike segments (Corollary 1). Let L_j be a list of the segments.

Step 4. Pick a treelike segment S from L_j; if none remain, eliminate A_j and go to Step 1. Use $T(u)$ and $T(w)$ and d_i, d_i' for $i = 1, \ldots, n$ to locate a shortest-path tree $T(S)$ in G rooted at S.

Step 5. If not using Technique 2, determine whether the point x_s that solves (6) lies on A_j; if so, let $l = j$ and go to Step 8; if not, remove S from L_j and go to Step 4. If using Technique 2, find the arc $A_l = A[\hat{u}, \hat{w}]$ on which the solution x_s of (18) lies. Do so by following a path along which $z^T(y)$ decreases, starting with A_j, until x_s is bracketed on an arc A_l. Eliminate all arcs (if any) in $T(S)$ between A_j and A_l, excluding A_j and A_l. If $j = l$, go to Step 8.

Step 6. Remove from L_j the segments of A_j that lie between S and A_l on $T(S)$, including S.

Step 7. Compute $\hat{d} = d(\hat{u}, v_i)$ and $\hat{d}_i' = d(\hat{w}, v_i)$ for $i = 1, \ldots, n$, and compute $\hat{s}_i = \frac{1}{2}[\hat{d}_i' - \hat{d}_i + b(\hat{u}, \hat{w})]$. Perform Step 2 with A_l replacing A_j, and branch to Step 1 if $z_0 \geq z_{\min}$. If $z_0 < z_{\min}$, perform Step 3 with A_l and $\hat{s}_i$ replacing A_j and s_i, resulting in list L_l.

Step 8. Find the point x_s on A_l. Do so by solving (26) if using Technique 2, and by solving (8) otherwise. If $l \neq j$, go to Step 10.

Step 9. If $x_s \in S$, set $k = k + 1$, $x_k^* = x_s$, and $z_{\min} = \min\{z_{\min}, z(x_k^*)\}$, remove S from L_j, and go to Step 4. Otherwise do the following. If not using Technique 2, remove S from L_j and go to Step 4. If using Technique 2, remove from L_j the segments of A_j

that lie between x_s and S, including S but excluding the segment containing x_s. Then go to Step 4.

Step 10. Remove from L_l the segments of A_l (if any) that lie between x_s and S in $T(S)$, excluding the segment containing x_s. Go to Step 4.

Step 11. Any x_j^* for which $z(x_j^*) = \min_{1 \leqslant l \leqslant k}\{z(x_l^*)\}$ solves (1).

6. Semiseparable Functions

In this section, we state without proof several properties of semiseparable functions that we demonstrate elsewhere (1984, 1985). The intent is to show that a large class of useful convex cost functions are nondecreasing and semiseparable and hence satisfy the conditions of Theorem 3.

Although convex combinations of semiseparable functions are not in general semiseparable, convex combinations of "strongly" semiseparable functions are strongly semiseparable. Let R^{n+} be the set of all points $(x_1, \ldots, x_n) \in R^n$ satisfying $x_i \geqslant 0$ for $i = 1, \ldots, n$. We say that $f\colon R^{n+} \to R$ is *strongly semiseparable* if, for all permutations π, the function

$$f_\pi(x+\beta\Delta x, y+\alpha 1_t, z) - f_\pi(x+\alpha 1_s+\beta\Delta x, y, z) \quad (29)$$

is nonincreasing with respect to $\beta \geqslant 0$ for all $x, \Delta x \in R^{s+}$, $y \in R^{t+}$, $z \in R^{u+}$, and all integers $s, t, u \geqslant 0$ with $s + t + u = n$, where $\alpha \geqslant 0$. It is clear that strongly semiseparable functions are semiseparable.

Let $f^0(z; v)$ denote the generalized directional derivative of $f\colon D \subseteq R^{n+} \to R$ with respect to vector $v \in R^n$ (as defined in Clarke 1975). It is possible to show that f is strongly semiseparable if for any permutation π, $f_\pi^0(x, y, z; 1_s, -1_t, 0_u)$ is nondecreasing with respect to each x_j, $j = 1, \ldots, s$, for all $x \in R^{s+}$, $y \in R^{t+}$, $z \in R^{u+}$, and all integers $s, t, u \geqslant 0$ with $s + t + u = n$. This result implies that the unweighted maximum function defined on R^{n+}, as well as any convex twice-differentiable function on R^{n+} whose Hessian is a diagonally dominant matrix throughout R^{n+}, is strongly semiseparable. Since the Hessian of an additively separable function is a diagonal and hence a diagonally dominant matrix, any convex additively separable function on R^{n+} is strongly semiseparable.

Several well-known location problems have convex, nondecreasing and semiseparable cost functions. The 1-median and weighted 1-center problems have cost functions that are respectively linear and weighted maximum functions and are therefore semiseparable, as well as convex and nondecreasing. Cent-dian problems in which cost is a convex combination of a linear and an unweighted maximum function, both strongly semiseparable, have strongly semiseparable, convex and nondecreasing cost functions. This convex combination is not in general semiseparable, however, if the maximum function is weighted, since the latter is not strongly semiseparable. Problems in which cost is an L_p norm of distances with $p \geqslant 1$ (such as the center of gravity problem, with $p = 2$) can be solved by minimizing the p-th power of cost, which is strongly semiseparable (as well as convex and nondecreasing) because it is additively separable.

7. Side Constraints

Suppose that a solution $x \in G$ of (1) must satisfy a set of constraints of the form

$$g_k(d(x, v_1), \ldots, d(x, v_n)) \geqslant 0, \quad k = 1, \ldots, K. \quad (30)$$

We solve (1) as before, except that (6) or (18) must be solved with constraints (30). The line search problem (8) must therefore be solved with the additional constraints,

$$g_k(d(u, v_1) + t, \ldots, d(u, v_j) + t,$$
$$d(w, v_{j+1}) + b - t, \ldots, d(w, v_n) + b - t) \geqslant 0,$$
$$k = 1, \ldots, K, \quad (31)$$

where the nodes are indexed as for (9). Similarly, the line search problem (26) (used with Technique 2) must be solved with the additional constraints,

$$g_k(d^T(\hat{u}, v_1) + t, \ldots, d^T(\hat{u}, v_j) + t,$$
$$d^T(\hat{w}, v_{j+1}) + \hat{b} - t, \ldots, d^T(\hat{w}, v_n) + \hat{b} - t) \geqslant 0,$$
$$k = 1, \ldots, K, \quad (32)$$

where the nodes are indexed as for (28).

If each g_k in (30) is convex, the region satisfying (31) or (32) is convex and is therefore a (possibly empty) subinterval of $[b(u, x_1), b(u, x_2)]$ or $[0, \hat{b}]$, respectively. Thus if the constraint functions are convex, (2) can still be solved via the solution of line search problems, each of which is a nonlinear program with a convex feasible region and convex objective function in one variable.

8. Worst-Case Performance

In this section we determine the worst-case performance of the algorithm of Section 5. We first establish an upper bound on the order of its complexity, and we then show that the upper bound can be achieved in at least one instance.

Suppose we are to solve the unconstrained version of (2) (i.e., $D = G$) on a network of n nodes and m arcs. In the worst case we try to eliminate segments using Techniques 1 and 2, and eliminate none. Thus we solve (18) for every treelike segment S. By Corollary 2, there are at most $m(n-1) = O(mn)$ such segments.

The complexity of finding x_s on $T(S)$ can be analyzed as follows. Since no segments are eliminated, x_s must lie on either the arc A containing S or on an adjacent arc. (The latter is possible if the path from x_s to S does not contain an entire treelike segment of A or the arc on which x_s lies.) Since z^T is convex on $T(S)$, to determine on which arc x_s lies requires at most that z^T be evaluated twice (to determine whether z^T is increasing or decreasing) at either end of the $n-1$ or fewer arcs incident to a node of A, for a total of $O(n)$ function evaluations. If we are given a shortest-path tree rooted as S, each evaluation of $z^T(y)$ requires computation of $d_i^T = d^T(y, v_i)$ for $i = 1, \ldots, n$, a task of complexity $O(n)$ (as shown in Hooker 1984), and evaluation of $f(d_1^T, \ldots, d_n^T)$. Thus if the evaluation of $f(d_1^T, \ldots, d_n^T)$ has complexity $O(g(n))$, the evaluation of $z^T(y)$ is $O(n + g(n))$. Finding x_s on the arc containing it can be done at worst by bisection search. This computation requires that $z^T(y)$ be evaluated a number of times that is a logarithmic function of the ratio of the arc length to the desired precision and is therefore independent of n and m. Thus the complexity of finding x_s on $T(S)$ is $O(n(n + g(n)))$.

If Technique 2 is not used, one solves (6) rather than (18). Since no arcs other than A are involved, the complexity is $O(n + g(n))$.

Finally, computing shortest-path trees at either end of an arc has complexity of $O(n^2)$ using, say, the Dijkstra (1959) algorithm. So the total complexity of computing m shortest-path trees is $O(mn^2)$. (If all shortest paths are computed in advance, the complexity is of lower order, but more storage is required for the $n \times n$ array of distances.) The complexity of the sorting operation in Step 3 and the upper bound test for Technique 1 is dominated by that of computing shortest-path trees.

Since there are at most $O(mn)$ segments on which to solve the subproblems of complexity of at most $O(n(n + g(n)))$ or $O(n + g(n))$, we have proved the following result.

Lemma 8. *If the complexity of evaluating f at one point is $g(n)$, the complexity of the algorithm of Section 5 is no more than $O(mn^2 + mn^2(n + g(n))) = O(mn^2(n + g(n)))$ if Technique 2 is used and no more than $O(mn^2 + mn(n + g(n))) = O(mn(n + g(n)))$ otherwise.*

In the common case of $g(n) = O(n)$, the complexity bounds are simply $O(mn^3)$ with Technique 2 and $O(mn^2)$ without.

We now exhibit a problem, which we call Problem W, in which the bounds of Lemma 8 are achieved. Let Problem W be set on a complete network G_W of n nodes and $m = \frac{1}{2}n(n-1)$ arcs. List the arcs $e_1, \ldots, e_m$ in any order and let the length b_t of arc e_t be $2^M + 2^t$, where M is large. Let the objective function z be the median function, with each node assigned a weight of one.

Lemma 9. *If $M \geq m - 1$, network G_W has $m(n-1)$ treelike segments.*

Proof. Let $b_{ij} = b(v_i, v_j)$ be the length of arc (v_i, v_j), and let $d_{ij} = d(v_i, v_j)$. Recall from Corollary 2 that node v_k generates a segment boundary on the interior of (v_i, v_j) if $0 < s_k < b_{ij}$, where $s_k = \frac{1}{2}[d_{jk} - d_{ik} + b_{ij}]$, or equivalently,

$$-b_{ij} < d_{jk} - d_{ik} < b_{ij}. \tag{33}$$

Note that $b_{ij} \leq b_{ik} + b_{kj}$ for all i, j, k, so that $d_{ij} = b_{ij}$. Thus since $-b_{ij} < b_{jk} - b_{ik} < b_{ij}$ for all distinct i, j, k, (33) holds for all distinct i, j, k. Thus the interior of every arc (v_i, v_j) contains a segment boundary for each $v_k \neq v_i, v_j$. Since $b_{jk} - b_{ik}$ is different for every $k \neq i, j$, $d_{jk} - d_{ik}$ is different for every k and all $n - 2$ segment boundaries on (v_i, v_j) are distinct. Thus G_W has $m(n-1)$ treelike segments.

We have proved the following lemmas elsewhere (Hooker 1984, Lemmas 2.13, 2.14).

Lemma 10. *For sufficiently large M, Technique 1 eliminates no arcs in problem W.*

Lemma 11. *For sufficiently large M, the segments of G_W can be enumerated in such a way that Technique 2 eliminates none of them in problem W.*

From Lemma 9, we know that for sufficiently large M, G_W has $m(n-1)$ treelike segments, and from Lemmas 10 and 11, we know that all the segments S of G_W must be enumerated. Since the median on each $T(S)$ lies at one of the nodes incident to the arc A containing S, it is necessary to evaluate z^T on each of the $n-1$ arcs incident to this node, so as to establish that $z^T(y)$ is in fact nondecreasing as y moves away from the node in every direction. Again, finding a median on any $T(S)$ has complexity $O(n(n + g(n)))$. If Technique 2 is not used, solving (6) on S again has complexity $O(n + g(n))$. We have proved the following result.

Theorem 4. *The complexity bounds of Lemma* 8 *can be achieved.*

Although the use of Technique 2 increases worst-case complexity, we will see in the next section that its use in fact reduces computation time on the several random and real-world networks on which it has been tested.

9. Computational Experience

A FORTRAN implementation of the algorithm of Section 5 was tested for efficiency both on random networks and on actual highway, rail and waterway networks. This section reports the results.

All testing was carried out on a DECsystem-10 computer running under the TOPS-10 monitor. The FORTRAN code was compiled by Version 6 of the FORTRAN-10 compiler. The code is a research code written for flexibility rather than efficiency. Note that CPU times are in seconds.

The computer program was first tested on random networks, generated as follows. The size of the network was set at n nodes and m arcs; m numbers uniformly distributed between zero and 10 were then randomly assigned positions in the upper triangle of an $n \times n$ distance matrix (diagonal excluded). These represented the arcs and their lengths. If the arcs covered fewer than n nodes, or if the network was disconnected, it was discarded and another generated. Finally, weights uniformly distributed between zero and one were assigned to the nodes.

Table I displays the performance of the computer code on ten random networks. In the top half of the table the cost function is the center-of-gravity function, given by $f(d_1, \ldots, d_n) = \sum_{i=1}^{n} w_i d_i^2$, with each $w_i \geq 0$. In the bottom half, the objective function is a cent-dian function in which the 1-center component is given "corrected" weight $\beta \equiv \gamma z_c^*/[\gamma z_c^* + (1 - \gamma) z_m^*] = 1/2$, z_c^* and z_m^* are the optimal values of the 1-center and 1-median functions on the network in question, and γ is the actual weight given the 1-center component. Since typically $z_m^* \gg z_c^*$, β is a much better indication of the relative importance of the two components than γ.

There is reason to believe that Technique 1, at least, would be more effective on real-world networks. An arc near the periphery of a network is generally more

Table I
Computational Results from 10 Random Networks

Network	Number of		Basic Algorithm		Percent of Segments Examined and CPU Seconds, Using Elimination Technique					
					No. 1 Only		No. 2 Only		Nos. 1 and 2	
	Nodes	Arcs	Treelike Segments	CPU	%	CPU	%	CPU	%	CPU
Center of Gravity										
1	20	40	270	4	80	3	62	3	55	3
2	40	80	846	21	77	17	59	16	44	12
3	60	120	1845	73	71	54	46	43	30	29
4	80	160	3153	176	70	128	46	100	31	68
5	100	200	3972	265	48	140	45	167	25	105
6	40	60	384	11	63	7	53	9	37	7
7	40	80	846	21	77	17	59	16	44	12
8	40	120	1276	33	85	29	44	23	40	21
9	40	160	2521	61	87	54	50	38	44	34
10	40	200	3366	144	92	77	49	49	44	45
Cent-Dian										
1	20	40	270	4	76	3	69	3	51	3
2	40	80	846	20	59	13	60	16	34	10
3	60	120	1845	71	32	28	45	43	14	17
4	80	160	3153	207	51	118	42	98	17	48
5	100	200	3972	263	54	158	45	164	21	93
6	40	60	384	11	47	7	51	10	21	6
7	40	80	846	20	59	13	60	16	34	10
8	40	120	1276	36	74	28	50	26	41	23
9	40	160	2521	66	83	57	52	43	44	37
10	40	200	3366	89	81	69	53	54	47	48

Table II
Computational Results for 2 Rail, 2 Waterway and 3. U.S. Interstate Highway Networks

Networks[a]	Number of		Basic Algorithm		Percent of Segments Examined and CPU Seconds, Using Elimination Technique					
					No. 1 Only		No. 2 Only		Nos. 1 and 2	
	Nodes	Arcs	Treelike Segments	CPU	%	CPU	%	CPU	%	CPU
Center of Gravity										
1	116	122	196	29	89	29	21	17	21	17
2	66	72	128	9	66	7	44	6	33	6
3	130	162	447	70	11	33	85	65	8	32
4	55	69	155	7	76	7	43	6	39	6
5	109	152	878	99	21	49	65	84	13	45
6	175	280	2186	350	17	142	50	265	8	123
7	276	432	4264	—[b]	—	—	—	—	6	414
Cent-Dian										
1	116	122	196	30	22	21	46	22	21	17
2	66	72	128	9	32	6	52	7	17	5
3	130	162	447	63	18	34	86	60	17	34
4	55	69	155	7	77	6	40	6	33	6
5	109	152	878	96	19	44	68	79	12	41
6	175	280	2196	329	15	127	53	250	9	121
7	276	432	4264	—	—	—	—	—	7	395

[a] The networks are: 1, Denver and Rio Grande Western Railroad; 2, Duluth, Missabe and Iron Range Railway; 3, U.S. coastal merchant marine shipping lanes; 4, Great Lakes shipping lanes; 5, U.S. Interstate highways west of the Mississippi (except Alaska and Hawaii), including cities along the Mississippi (except New Orleans); 6, U.S. Interstate highways east of the Mississippi, including cities along the Mississippi; and) 7, U.S. Interstate highways in the continental 48 states.
[b] Computation omitted because of excess CPU time.

likely to be eliminated than one nearer the middle of a network. Random networks, unlike most actual networks, have no clear periphery.

Table II confirms this expectation. The railroad arc lengths are impedances based on travel distance and railroad class (Peterson 1984). The waterway arc lengths are travel distances. All nodes receive equal weight in the rail and waterway networks. The Interstate Highway networks are taken from a road atlas (Rand McNally 1984). The arc lengths are travel distances, and each node is weighted by the population of the city (if any) at which it is located. Since they may be of interest, the solutions of the Interstate Highway problems are displayed in Table III.

Table III
Center of Gravity and Cent-Dians for 3 U.S. Interstate Highway Networks

Network	Center of Gravity	Cent-Dians
1. U.S. west of the Mississippi	Las Cruces, NM (intersection of I-10 and I-25, 12 miles north of El Paso, TX, on I-25)	$0 \leq \beta \leq 0.25$ (median): Las Cruces, NM
		$0.25 \leq \beta \leq 0.33$; Denver, CO
		$0.33 \leq \beta \leq 1$ (center): 12.5 miles east of Denver on I-70
2. U.S. east of the Mississippi	Cambridge, OH (intersection of I-70 and I-77, 79 miles east of Columbus on I-70)	$0 \leq \beta \leq 0.08$ (median): Harrisburg, PA
		$0.08 \leq \beta \leq 0.14$: Washington, PA (25 miles south of Pittsburgh)
		$0.14 \leq \beta \leq 0.64$: Cambridge, OH
		$0.64 \leq \beta \leq 0.94$: 28.5 miles south of Cambridge, OH, toward Charleston, WV, on I-77
		$0.94 \leq \beta \leq 1$ (center): 48.5 miles north of Charleston, WV, toward Pittsburgh on I-79
3. The continental 48 states	St. Louis, Mo	$0 \leq \beta \leq 0.12$ (median): Indianapolis, IN
		$0.12 \leq \beta \leq 0.20$: intersection of I-57, and I-70, 80 miles SE of Springfield, IL, 138 miles west of Indianapolis toward St. Louis on I-70
		$0.20 \leq \beta \leq 0.39$: St. Louis, MO
		$0.39 \leq \beta \leq 0.46$: Topeka, KS
		$0.46 \leq \beta \leq 1$ (center): 38.5 miles east of Salina, KS, on I-70

Acknowledgment

My thanks to Robert Garfinkel for helpful suggestions and astute criticism. Also, Timothy Lowe suggested the use of generalized directional derivatives in Section 6.

References

CHURCH, R. L., AND R. S. GARFINKEL. 1978. Locating an Obnoxious Facility on a Network. *Trans. Sci.* **12,** 107–117.

CLARKE, F. H. 1975. Generalized Gradients and Applications. *Transact. Am. Math. Soc.* **205,** 247–262.

DEARING, P. M. 1977. Minimax Location Problems with Nonlinear Costs. *J. Res. Natl. Bur. Stand.* **82,** 65–72.

DEARING, P. M., AND R. L. FRANCIS. 1974. A Minimax Location Problem on a Network. *Trans. Sci.* **8,** 333–343.

DEARING, P. M., R. L. FRANCIS AND T. J. LOWE. 1976. Convex Location Problems on Tree Networks. *Opns. Res.* **24,** 628–642.

DIJKSTRA, E. W. 1959. A Note on Two Problems in Connexion with Graphs. *Number. Math.* **1,** 269–271.

FRANCIS, R. L. 1977. A Note on a Nonlinear Minimax Location Problem in Tree Networks. *J. Res. Natl. Bur. Stand.* **82,** 73–80.

GOLDMAN, A. J. 1971. Optimal Center Location in Simple Networks. *Trans. Sci.* **5,** 212–221.

GOLDMAN, A. J., AND P. M. DEARING. 1975. Concepts of Optimal Location for Partially Obnoxious Facilities. *Bull. Opns. Res. Soc. Am.* **23,** Supplement 1, B-31.

HAKIMI, S. 1964. Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph. *Opns. Res.* **12,** 450–459.

HAKIMI, S. L. 1965. Optimum Distribution of Switching Centers in a Communications Network and Some Related Graph-Theoretic Problems. *Opns. Res.* **13,** 462–475.

HALPERN, J. 1976. The Location of a Center-Median Convex Combination on an Undirected Tree. *J. Reg. Sci.* **16,** 237–245.

HALPERN, J. 1978. Finding Minimal Center-Median Convex Combinations (Cent-Dian) of a Graph. *Mgmt. Sci.* **24,** 535–544.

HALPERN, J. 1980. Duality in the Cent-Dian of a Graph. *Opns. Res.* **28,** 722–735.

HANDLER, G. Y. 1973. Minimax Location of a Facility in an Undirected Tree Graph. *Trans. Sci.* **7,** 287–293.

HANDLER, G. Y., AND P. B. MIRCHANDANI. 1979. *Location on Networks: Theory and Algorithms.* MIT Press, Cambridge, Mass.

HOOKER, J. N. 1984. Nonlinear Network Location Models. Ph.D. thesis, University of Tennessee, Knoxville.

HOOKER, J. N. 1985. Solving Nonlinear Single-Facility Network Location Problems. Working paper 18-84-85, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pa.

LEVY, J. 1967. An Extended Theorem for Location on a Network. *Opnl. Res. Quart.* **18,** 433–442.

MINIEKA, E. 1983. Anticenters and Antimedians of a Network. *Networks* **13,** 359–364.

PETERSON, B. E. 1984. *Interline, a Railroad Routing Model: Program Description and User's Manual.* Oak Ridge National Laboratory Technical Monograph ORNL/TM-8944, Oak Ridge, Tenn.

Rand McNally. 1984. *Road Atlas.* Rand McNally, New York.

SHIER, D. R., AND P. M. DEARING. 1983. Optimal Locations for a Class of Nonlinear, Single-Facility Location Problems on a Network. *Opns. Res.* **31,** 292–303.

TANSEL, B. C., R. L. FRANCIS, T. J. LOWE AND M. L. CHEN. 1982. Duality and Distance Constraints for the Nonlinear p-Center Problem and Covering Problem on a Tree Network. *Opns. Res.* **30,** 725–743.