EQUIVALENT MATHEMATICAL PROGRAMMING FORMULATIONS OF MONOTONIC TREE NETWORK LOCATION PROBLEMS

E. ERKUT
University of Alberta, Edmonton, Canada

R. L. FRANCIS
University of Florida, Gainesville, Florida

T. J. LOWE
Purdue University, West Lafayette, Indiana

A. TAMIR
New York University, New York and Tel Aviv University, Tel Aviv, Israel

(Received October 1986; revisions received August 1987; February, May 1988; accepted May 1988)

We consider the optimization problem of locating several new facilities on a tree network, with respect to existing facilities, and to each other. The new facilities are not restricted to be at vertices of the network, but the locations are subject to constraints. Each constraint function, and the objective function, is an arbitrary, nondecreasing function of any finite collection of tree distances between new and existing facilities, and/or between distinct pairs of new facilities, and represents some sort of transport or travel cost. The new facilities are to be located so as to minimize the objective function subject to upper bounds on the constraint functions. We show that such problems are equivalent to mathematical programming problems which, when each function is expressed using only maximization and summation operations on nonnegatively weighted arguments, are linear programming problems of polynomial dimensions. The latter problems can be solved using duality theory with special purpose column generation and shortest path algorithms for column pricing.

Network location problems occur when new facilities must be located on a transport network of some sort, such as an air, aisle, highway, river or sea lane network. Existing facilities on the network have locations that can be represented by vertices. Travel between facilities results in costs of some sort. An objective function to be minimized represents cost, which is usually nondecreasing in the distances between facilities.

The network location problems we consider are continuous, in the sense that we allow new facilities to be located at any points of the network—not just at vertices. Similarly, our problems do not have fixed costs. We wish to find optimal locations: ones that minimize the objective function while satisfying certain constraints. These constraints may impose upper bounds on distances between facilities or, more generally, on nondecreasing functions of distances between facilities. In case the network is a tree, our approach leads to an equivalent mathematical programming problem which, for many cases of interest, is a linear program with a polynomial number of variables and constraints. For the nontree case, our approach can be used to obtain upper and lower bounding problems.

We are given a tree T, defined as in Dearing and Francis (1974b) with positive arc lengths. The tree has m vertices, $v_1, \ldots, v_m$, each of which is the location of some existing facility. We wish to locate n new facilities on the tree T, at locations denoted by $x_1, \ldots, x_n$, to be determined. With (nonnegative) tree distances defined in the usual way, we have a collection of distance functions; $d(x_j, v_i)$ is the distance between new facility j and existing facility i, while $d(x_j, x_k)$ is the distance between new facilities j and k. Letting q denote the cardinality of the subset of distance functions of interest, we note that

$$q \leq m\,n + (n-1)n/2. \tag{1}$$

With $X = (x_1, \ldots, x_n)$ the vector of new facility locations, we let $D(X)$ denote a vector of all the

Subject classifications: Facilities/equipment planning: location problems. Networks/graphs, applications: network location problems. Programming: linear programming applications.

0030-364X/89/3703-0447 $01.25

distance functions of interest, where the q entries in $D(X)$ occur in some well defined order.

We have an objective function, as well as constraints, which depend upon the tree distances between facilities. The problem of interest, denoted by **PM** for the monotonic problem, is as follows.

Problem PM

Minimize $f_0(D(X))$

subject to $f_k(D(X)) \leq b_k, \quad k = 1, \ldots, p.$

We assume each (real-valued) function f_k, $k = 0, 1, \ldots, p$, is nondecreasing in each component of $D(X)$.

We show that **PM** is equivalent to a mathematical programming problem. When each monotonic function f_k has a certain structure involving summation and/or maximization operations, **PM** is equivalent to a linear programming problem which is tractable using the revised simplex algorithm with special shortest path algorithms for column pricing and column generation. We briefly summarize some computational experience for a special linear case.

If we pose **PM** on a general, undirected and finite graph G with positive arc lengths and shortest path distances we get a difficult problem. Even without constraints the objective function includes the n-center problem, which is NP-hard (Hsu and Nemhauser 1979, Kariv and Hakimi 1979a), and the n-median problem, which is also NP-hard (Kariv and Hakimi 1979b). Results of Kolen (1982) imply that determining whether or not there exists a feasible solution (one satisfying the constraints) is NP-hard; Kolen also exhibits unconstrained "multimedian" and "multicenter" location problems which are NP-hard. Hence we concentrate on the case when G is a tree T. Having a tree allows us to formulate an equivalent linear program for many cases of interest.

Many location problems (e.g., all the location theory journal papers we reference except for the review papers) involve costs that are nondecreasing in distance, so that our monotonicity assumption is often quite reasonable. We observe, given any two nondecreasing functions, that the: 1) sum, 2) maximum, 3) minimum, and 4) composition of the two functions (whenever it is defined) is also a nondecreasing function. Furthermore, if both functions are always nonnegative, and their product is well defined, then 5) the product is also a nondecreasing function. Hence, many operations with nondecreasing functions of distance yield nondecreasing functions of distance, so that many tree network location problems can be put into the above form.

The literature on discrete location theory problems, which are solved by solving some linear programming relaxation, is vast (Francis and Mirchandani 1989). We emphasize that we are *not* studying such problems. Rather, we are studying location problems which are essentially continuous in nature, and we obtain equivalent mathematical programs, *not* relaxations.

While we know of no literature for the general version of our problem, the literature for various special cases is substantial. The literature that we know of related to our problem is for the unconstrained case when f_0 is linear, for the constrained case when $n = 1$, and for certain minimax location problems which we will discuss. Almost all the literature we cite considers the case of a tree network.

For the case of no constraints, the problem where the objective function is a sum of (weighted) distances has been solved by Picard and Ratliff (1978) and Kolen (1981, 1982); the "multicenter" problem where the objective function is a maximum of (weighted) distances has been solved by Francis, Lowe and Ratliff (1978) (abbreviated henceforth as FLR).

The literature on network location problems with distance constraints is not large, with the exception of covering problems (see Kolen and Tamir 1989). Halpern (1976, 1978, 1980) studied locating a single new facility subject to upper bounds on its distance to existing facilities, as did Handler (1985). Dearing, Francis and Lowe (1976) formulated a number of problems with distance constraints and proved they are convex problems, but gave no algorithms except for $n = 1$. FLR, who introduced the so-called separation conditions we exploit subsequently, studied whether or not a solution exists for collections of distance constraints, and applied their results to solve a minimax problem (mentioned above); Tansel, Francis and Lowe (1980) build on their work to solve some multiobjective location problems. Tansel, Francis, Lowe and Chen (1982) considered distance constraints between existing facilities and closest new facilities for the p-center problem. Moon and Chaudhry (1984) discuss applications of location problems with distance constraints, suggest a classification scheme for such problems, and report experience with some integer programming formulations.

There is some literature on a minimax multifacility location problem with rectilinear distances solved by Dearing and Francis (1974a) which includes distance constraints. Often a location problem with rectilinear distances can be interpreted as a sequence of location problems on line segments. We can consider each line segment to be a tree that is a path. Such problems on

paths often have equivalent linear programming formulations which can be constructed quite directly (Francis and White 1974). Dearing and Langford (1975) showed that some tree network location problems could be converted to equivalent rectilinear distance location problems, which then could be solved using linear programming.

Besides the literature on covering problems, there is also some other literature on duality related to our work. Specifically, see Dearing and Francis (1974b), Chan and Francis (1976), Dearing (1977), Francis (1977), FLR, Halpern (1980), Tansel et al. (1982), and Kolen (1982).

In the next section, we show that problem **PM** is equivalent to the following, where Z is a vector of real variables:

$$\begin{aligned} &\text{minimize} && f_0(Z) \\ &\text{subject to} && f_k(Z) \leqslant b_k, \quad k = 1, \ldots, p. \\ & && \mathrm{D}(X) \leqslant Z. \end{aligned}$$

In FLR, it was established that the constraints $\mathrm{D}(X) \leqslant Z$ can be replaced by "separation conditions," a collection of equivalent linear inequalities $A\,Z \geqslant d$, which, together with $Z \geqslant 0$, gives a mathematical programming problem. Given an optimal solution Z^* to the latter problem, an $O(n(m+n))$ algorithm they call the sequential location procedure and term SLP, then can be used to construct new facility locations X^* which satisfy $\mathrm{D}(X^*) \leqslant Z^*$, so that X^* solves the original problem, **PM**.

It is our reliance on the separation conditions that requires the assumption that the network on which the facilities are to be located is a tree. If the network is not a tree, it is known that the separation conditions are only necessary conditions for the constraints $\mathrm{D}(X) \leqslant Z$ to be satisfied. Consequently, for a general network the solution of the mathematical programming problem we shall construct provides a lower bound on the minimum objective function value of the original problem. Even if this lower bound is attained, it can be difficult to construct the corresponding locations given the distances; Kolen (1982) has proven that such a problem is NP-hard.

Suppose we formulate **PM** on a general network G as, say, (**PM**: G), using shortest path distances, and denote the minimum objective function value of (**PM**: G) by $f_0^*(G)$. We can get an upper bound on $f_0^*(G)$ by solving **PM** on any spanning *tree* for which a feasible solution exists, using the approaches we shall develop. Hence ways exist of computing both upper and lower bounds on $f_0^*(G)$. In follow-on research to ours, Erkut, Francis and Lowe (1988) developed some of these ideas, obtaining computational experience by computing upper and lower bounds on minimum objective function values for a number of "multimedian" problems; the average gap between the bounds was approximately 4%.

It has been the case that most tree network location problems were solved on an ad hoc basis, with a new solution procedure being devised for each problem. The advantage of our mathematical programming model is that we obtain a unifying collection of theory applicable to many problems. Due to the very large body of mathematical programming research, the opportunities for knowledge transfer appear quite substantial. For example, we shall use duality theory, column generation, and the revised simplex method. These are all well known linear programming tools that have not been used previously for the location problems we consider. Likewise, it will now be possible to apply mathematical programming results on sensitivity analysis and parametric programming to the location problems we study. Furthermore, our model is sufficiently general to include, as special cases, tree network location models which have never been studied; in Section 2 we give an example of one such model. We believe it will be possible to use our approach as a means of obtaining insight into problems previously too complicated to consider. Once such insight is obtained we suspect it will then be possible to devise algorithms which improve upon our approach; certainly previous study of combinatorial problems with equivalent linear programming formulations (see, for example, Grötschel, Lovász and Schrijver 1981) suggests that such should be the case.

In a survey paper, Tansel, Francis and Lowe (1983) raise the question of why tree network location problems are more tractable than location problems on general networks, and point out that convexity results (Dearing, Francis and Lowe) constitute a partial explanation. One can also point to NP-completeness results for general networks of Hsu and Nemhauser (1979), Kariv and Hakimi (1979a, b), Kolen (1982), to equivalent linear programming formulations of covering location problems on tree networks (Kolen and Tamir 1989), and to graph theory (Kolen 1982). We think our obtaining equivalent mathematical programming problems is a further explanation. Many well solved tree network location problems have equivalent formulations as linear programming problems.

We now give an overview of the remainder of this paper. In Section 1 we obtain an equivalent mathematical programming problem. All results in Section

1 apply to problem **PM**, but those of Sections 2 and 3 require additional "convexity" assumptions. In Sections 2 and 3 we concentrate on equivalent linear programming problems; in this case, the f_k functions are linear or "isotone polyhedral convex" (defined in Section 2). For this case, the dual problem can be solved using the revised simplex method with column generation and special shortest path algorithms for column pricing. At the end of Section 3, we outline the computational experience for a special case of our problem which we call the *multimedian problem*, denoted as follows.

Problem PMM

Minimize $c^T \mathrm{D}(X)$

subject to $\mathrm{D}(X) \leq b, \quad X \in T^n$

(T^n is the n-fold Cartesian product of T with itself, as defined by Dearing, Francis and Lowe) where b is a given vector with positive entries and c has nonnegative entries. In Section 4 we give a numerical example of the multimedian problem. We recommend periodic reference to Section 4 to see the various ideas of Sections 1, 2 and 3 illustrated.

1. An Equivalent Mathematical Programming Problem

In this section, we obtain two mathematical programming problems that are equivalent to **PM**. In order to be precise, we first define an isotone function (see Reinboldt 1970; More and Reinboldt 1973). Let E_r^+ denote the nonnegative orthant of E_r (Euclidean r-space), and f be a function from E_r^+ into E_s. We say that f is *isotone* if for any $U, V \in \mathrm{E}_r^+$, $U \leq V$ implies $f(U) \leq f(V)$. Here inequalities are component-wise inequalities. Note that we do not require $s = 1$.

A fundamental result that we shall use is the following well known, easily proven lemma.

Monotonicity Lemma. *Let f be an isotone function from E_r^+ into E_s. For any $b \in \mathrm{E}_s$ and $D \in \mathrm{E}_r^+$, the following are equivalent.*

(a) We have $f(D) \leq b$.
(b) There exists $Z \in \mathrm{E}_r^+$ such that $D \leq Z, f(Z) \leq b$.

That (a) implies (b) is trivial, while (b) implies (a) uses the isotonicity of f.

From this point on we assume that each function f_k is isotone, $k = 0, 1, \ldots, p$. Then, an equivalent version of **PM** is clearly as follows:

minimize b_0

subject to

$f_k(\mathrm{D}(X)) \leq b_k, \quad k = 0, 1, \ldots, p.$

The Monotonicity Lemma thus gives the following equivalent constraints:

$f_k(Z) \leq b_k, \quad k = 0, 1, \ldots, p;$

$\mathrm{D}(X) \leq Z.$

If there is an optimal solution to this problem then there exists an optimal solution, say X^*, Z^*, with $Z^* = \mathrm{D}(X^*)$. Essentially, we shall make a change of variables in order to work with real variables instead of tree distances; monotonicity allows us to relax the constraints $Z = \mathrm{D}(X)$ to $\mathrm{D}(X) \leq Z$.

We formulate the latter problem as an equivalent mathematical programming problem by replacing $\mathrm{D}(X) \leq Z$ by an equivalent set of linear inequalities. We construct an undirected network $N(Z)$ as follows. (Figure 1 illustrates $N(Z)$ for $m = 4$, $n = 3$.) Let $N(Z)$ have nodes $E_1, \ldots, E_m$ corresponding to the existing facilities, as well as nodes $N_1, \ldots, N_n$ corresponding to the new facilities. Include an arc (E_i, N_j) corresponding to each distance function $d(x_j, v_i)$ in $\mathrm{D}(X)$; the arc length is the entry in Z corresponding to this distance. Include an arc (N_j, N_k) corresponding to each distance function $d(x_j, x_k)$ in $\mathrm{D}(X)$; the arc length is the entry in Z corresponding to this distance. We assume $N(Z)$ is connected, otherwise **PM** decomposes into independent problems corresponding to the components of $N(Z)$. Similarly, we assume that the subnetwork of $N(Z)$ induced by the N nodes is connected. Let $L'(\mathrm{E}_h, \mathrm{E}_i: Z)$ denote the length of a shortest

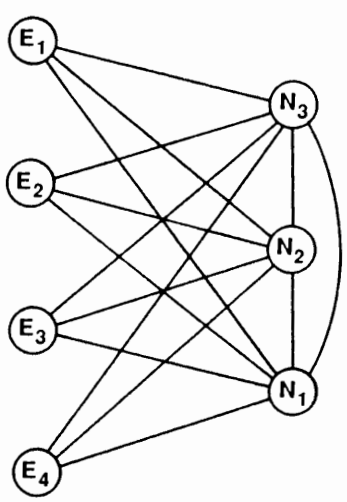

Figure 1. The network $N(Z)$ for the example of Section 4.

(simple) path between any two nodes E_h and E_i of $N(Z)$. FLR proved that the constraints $D(X) \leqslant Z$ are equivalent to the following inequalities, which they call the *separation conditions*:

$$L'(E_h, E_i: Z) \geqslant d(v_h, v_i), \quad 1 \leqslant h < i \leqslant m. \tag{2}$$

Also, FLR gave SLP, an $0(n(m+n))$ algorithm which, given Z, either finds an X for which $D(X) \leqslant Z$, or else determines that no such X exists. Because the order of SLP is the same as that of the data, it is known, given reasonable assumptions (Tansel, Francis and Lowe 1980) that SLP is a lowest order (worst case) algorithm for the problem it solves.

We call a path in $N(Z)$ between E_h and E_i *direct* if it is simple and contains exactly two E nodes, E_h and E_i. Tansel, Francis and Lowe (1980) proved that an equivalent version of the separation conditions is obtained if we replace the left-hand side of (2) by $L(E_h, E_i: Z)$, defined to be the length of a shortest direct path in $N(Z)$ between E_h and E_i, $1 \leqslant h < i \leqslant m$. Henceforth, we refer to the collection of inequalities involving shortest direct paths as the separation conditions.

Let A be a zero-one path-arc incidence matrix such that each row of A corresponds to some direct path in $N(Z)$, and each column corresponds to an arc of $N(Z)$. The ones in a row of A thus identify the arcs of $N(Z)$ in some direct path. Let d be a vector with each entry some tree distance, say $d(v_h, v_i)$. A and d have the same number of rows; for any row of A corresponding to a direct path between some E_h and E_i, the corresponding entry in d is $d(v_h, v_i)$. An equivalent version of the separation conditions (2) is that the length of *every* direct path between E_h and E_i (a linear function in Z) is at least $d(v_h, v_i)$. Hence, an equivalent form of the separation conditions, which we call the (direct) *path constraints*, is

$$AZ \geqslant d, \quad Z \geqslant 0. \tag{3}$$

Thus, given $Z \geqslant 0$, there exists $X \in T^n$ such that $D(X) \leqslant Z$ if and only if $AZ \geqslant d$.

Consequently, **PM** *is equivalent to the following mathematical programming problem*, which we denote by **PMP1**.

Problem PMP1

Minimize $f_0(Z)$

subject to

$f_k(Z) \leqslant b_k, \quad k = 1, \ldots, p,$

$AZ \geqslant d,$

$Z \geqslant 0.$

Given an optimal solution to **PMP1**, say Z^*, we can apply SLP to the network $N(Z^*)$ to construct a corresponding optimal solution, X^*, the locations of the new facilities, to **PM**.

The price we pay to obtain **PMP1** is that we have a real variable corresponding to *every* distance function in $D(X)$, as well as the extra linear inequalities $AZ \geqslant d$. Fortunately, the linear inequalities can be replaced by a smaller collection of linear inequalities, as our theorem of this section will establish.

When each function f_k is convex as well as isotone, then **PMP1** is a convex programming problem having mostly linear constraints, for which there exist well known optimality conditions and duality relations. For the (equivalent) problem (**PM**), we know of no such results. Beginning with Section 2, we shall concentrate on problems having equivalent linear programming formulations. We reserve equivalent convex programs for future study.

PMP1 may not appear too tractable, since there is a row of A for every direct path in $N(Z)$, and it is easy to show there can be as many as

$$C_{m,2} \sum_{j=1}^{n} j!\, C_{n,j}$$

direct paths (here $C_{n,j}$ denotes all combinations of n things taken j at a time). However, there is a column of A for every arc of $N(Z)$, and there are exactly q arcs, with q satisfying (1). Hence, it will usually be the case that A has many more rows than columns, a fact we exploit in our computational work.

Noting that the matrix A has $O(m^2 n!)$ rows, we give a result that justifies the replacement of the system $AZ \geqslant d$ by a system of inequalities with the same number of variables, but where the number of inequalities is *polynomial* in m and n.

Without loss of generality we assume (1) holds with equality, so that the network $N(Z)$ has every possible arc length, that is

$$z(E_i, N_j), \quad i = 1, \ldots, m \quad \text{and } j = 1, \ldots, n;$$

$$z(N_j, N_k), \quad 1 \leqslant j < k \leqslant n.$$

Also we define

$$z(N_j, E_i) = z(E_i, N_j), \qquad i = 1, \ldots, m; j = 1, \ldots, n.$$

$$z(N_k, N_j) = z(N_j, N_k), \quad 1 \leqslant j < k \leqslant n.$$

These are not extra variables, but simply alternative names for Z entries which we find convenient to use.

We say the vector Z *satisfies the type-EN triangle*

inequality conditions if

$$z(E_h, N_k) + z(N_k, N_j) - z(E_h, N_j) \geq 0,$$
$$1 \leq h \leq m; \quad 1 \leq j < k \leq n.$$

Likewise, we say that the vector Z *satisfies the type-NN triangle inequality conditions* if

$$z(N_j, N_p) + z(N_p, N_k) - z(N_j, N_k) \geq 0,$$
$$1 \leq j < k \leq n; \quad 1 \leq p \leq n, \quad p \neq j, k.$$

We say Z *satisfies the triangle inequality conditions* if Z satisfies both the type-EN and type-NN conditions. Note that the total number of type-EN triangle inequality conditions is $mn(n-1)/2$, while if $n \geq 2$ the total number of type-NN conditions is $n(n-1)(n-2)/2$.

Given any nodes E_h and E_i in $N(Z)$, let $S(E_h, E_i: Z)$ denote the length of a shortest direct path between E_h and E_i that uses *exactly one* N node; that is, $S(E_h, E_i: Z) = \min\{z(E_h, N_j) + z(N_j, E_i): j = 1, \ldots, n\}$.

We can now give our result.

Theorem. *There exists* $Z \geq 0$ *satisfying the separation conditions*

$$L(E_h, E_i: Z) \geq d(v_h, v_i) \quad \text{for } 1 \leq h < i \leq m \qquad (4)$$

if and only if there exists Z^*, $0 \leq Z^* \leq Z$, *which satisfies the triangle inequality conditions and*

$$S(E_h, E_i: Z^*) \geq d(v_h, v_i) \quad \text{for } 1 \leq h < i \leq m. \qquad (5)$$

Proof. ($\Rightarrow$) Define Z^* as follows: for each $1 \leq j < k \leq n$, let $z^*(N_j, N_k)$ be the shortest path length between N_j and N_k on the subnetwork of $N(Z)$ induced by the N nodes. For each E_h, N_j, $1 \leq h \leq m$, $1 \leq j \leq n$, define $z^*(E_h, N_j)$ to be the length of a shortest path in $N(Z)$ between E_h and N_j, over all paths connecting E_h and N_j and using *no* E node but E_h.

It is verified easily that Z^* satisfies the triangle inequality conditions. Also, $0 \leq Z^* \leq Z$ follows directly from the definition. Finally, observe that

$$S(E_h, E_i: Z^*) \geq L(E_h, E_i: Z) \quad \text{for } 1 \leq h < i \leq m$$

and thus (5) follows from (4).

($\Leftarrow$) It suffices to show that $L(E_h, E_i: Z^*) \geq d(v_h, v_i)$ for $1 \leq h < i \leq m$.

Consider some shortest direct path on $N(Z^*)$ connecting E_h and E_i. Let the path be characterized by the following ordered sequence of nodes: E_h, $N_{j(1)}, \ldots, N_{j(k)}$, E_i; that is, the path consists of E_h, E_i and the k intermediate N nodes. Thus

$$L(E_h, E_i: Z^*) = z^*(E_h, N_{j(1)}) + z^*(N_{j(1)}, N_{j(2)}) + \cdots + z^*(N_{j(k-1)}, N_{j(k)}) + z^*(N_{j(k)}, E_i).$$

If $k = 1$, then $j(1) = j(k)$ and

$$L(E_h, E_i: Z^*) = S(E_h, E_i: Z^*) \geq d(v_h, v_i).$$

Thus consider the case of $k \geq 2$; using the type-*NN* conditions repeatedly we get

$$L(E_h, E_i: Z^*) \geq z^*(E_h, N_{j(1)}) + z^*(N_{j(1)}, N_{j(k)}) + z^*(N_{j(k)}, E_i).$$

In order to use the type-EN conditions we must distinguish two cases: $j(1) < j(k)$; $j(1) > j(k)$. If $j(1) < j(k)$, then

$$z^*(N_{j(1)}, N_{j(k)}) + z^*(N_{j(k)}, E_i) \geq z^*(N_{j(1)}, E_i).$$

Thus

$$L(E_h, E_i: Z^*) \geq z^*(E_h, N_{j(1)}) + z^*(N_{j(1)}, E_i) \geq S(E_h, E_i: Z^*) \geq d(v_h, v_i).$$

If $j(1) > j(k)$ then

$$z^*(E_h, N_{j(1)}) + z^*(N_{j(1)}, N_{j(k)}) \geq z^*(E_h, N_{j(k)}).$$

Thus

$$L(E_h, E_i: Z^*) \geq z^*(E_h, N_{j(k)}) + z^*(N_{j(k)}, E_i) \geq S(E_h, E_i: Z^*) \geq d(v_h, v_i).$$

This completes the proof.

Let us denote by

$$\alpha Z \geq \delta$$

the triangle inequality constraints together with the path constraints

$$z(E_h, N_j) + z(N_j, E_i) \geq d(v_h, v_i)$$
$$1 \leq h < i \leq m, \quad 1 \leq j \leq n.$$

Each entry in α is -1, $+1$, or 0; each entry in δ is 0 or a distance between two tree vertices. We now have the following corollary.

Corollary. *For the problem* **PMP1**, *because each function* f_k *is isotone, we can replace the linear system* $A Z \geq d$ *by* $\alpha Z \geq \delta$ *and get an equivalent problem that we denote by* **PMP2**.

Thus (for $n \geq 2$) we replace a factorial number of constraints by

$$n(n-1)(n-2)/2 + m\,n(n-1)/2 + m(m-1)n/2$$
$$= n(n-1)(n-2)/2 + m\,n(m+n-2)/2$$

constraints. In reducing the number of constraints we have not altered the number of variables. This number of constraints, while polynomial in m and n, can still be large; for example, if $m = 50$ and $n = 10$, we get 14,860 constraints.

Finally, we consider the applicability of ellipsoidal algorithms (Bland, Goldfarb and Todd 1981). For the formulation **PMP1**, if the function f_k is linear for $k = 0, 1, \ldots, p$, then **PMP1** is a linear program with a nonpolynomial number of constraints. However, due to the existence of SLP, given a vector Z, we can test in (strongly) polynomial time whether Z is feasible, or else produce a violating constraint. Using the polynomial equivalence between strong separation and strong linear optimization (e.g., Lovász 1986), we conclude that the linear version of **PMP1** is solvable in polynomial time by ellipsoidal algorithms. If the functions f_k, $k = 0, 1, \ldots, p$ are convex and given by certain "oracles," then ellipsoidal algorithms can give an ϵ-approximation solution in time that is polynomial in terms of $|\log \epsilon|$ and the "complexity" of the oracles (Lovász).

Turning to the formulation **PMP2** and assuming that all functions are linear, we obtain a linear program of polynomial dimensions. In this case, in addition to ellipsoidal algorithms, we also can apply the (polynomial) projective methods pioneered by Karmarkar (1984). Unlike ellipsoidal algorithms, the latter methods seem to perform quite reasonably when compared with classical simplex type procedures.

2. The Linear Case: Polyhedral Convex Functions

In the spirit of Halpern's work (1976, 1978, 1980) on the cent-dian problem, consider the following *generalized cent-dian problem*, where $\mu > 0$, $D_h(X)$ denotes entry h of $D(X)$, and the w_h and ω_h are known nonnegative constants, "weights." The objective function is equivalent to a convex combination of a minisum and a minimax objective function.

Minimize

$$\sum_h w_h D_h(X) + \mu \max_h \{\omega_h D_h(X)\}$$

subject to $D(X) \leq b$.

The Monotonicity Lemma implies that the problem is equivalent to

$$\text{minimize} \quad \sum_h w_h z_h + \mu \max_h \{\omega_h z_h\}$$

$$\text{subject to} \quad D(X) \leq Z \leq b.$$

Since $\mu > 0$ we have, also equivalently,

$$\text{minimize} \quad \sum_h w_h z_h + \mu y$$

$$\text{subject to} \quad \omega_h z_h \leq y \quad \text{for all } h, \quad D(X) \leq Z \leq b.$$

Replacing the inequalities $D(X) \leq Z$ by either $AZ \geq d$ or $\alpha Z \geq \delta$, gives an equivalent linear programming problem solvable by the approach we shall develop.

When $n = 1$ (one new facility) the "max" term in the objective function is a "center" type term ($n = 1$ is the problem studied by Halpern). When $n \geq 2$, the max term is *not* equivalent to an "n-center" type term, but to a "multicenter" n-facility minimax problem with mutual communication (FLR). For reasons explained later, our approach will not handle "center-type" problems with more than one center.

We wish to generalize from this example. We observe that the objective function can be written as the composition of two isotone functions, say $f(Z) = g_1(g_0(Z))$, where $Z = D(X)$, $g_0(Z)$ is a vector with two entries, $y_1 = \Sigma\{w_h z_h : h\}$ and $y_2 = \max\{\omega_h z_h : h\}$, and $g_1(Y) = y_1 + \mu y_2$.

Thus consider the function f, where $f(Z) = g_s(\ldots g_1(g_0(Z))\ldots)$ is the composition of $s + 1$ functions, and is defined for $Z \geq 0$. Each function g_k is a mapping from $E_{n(k)}$ into $E_{n(k+1)}$ for some $n(k)$, $n(k+1)$, with $n(s+1) = 1$. In this section, we consider the case where each entry i in $g_k(Z_k)$ is of the form

$$\max_j \{a_{ijk}^T Z_k + b_{ijk}\}$$

where the maximum is over a finite number of terms (possibly only one, in which case entry i of $g_k(Z_k)$ is a linear function), each term a_{ijk}^T is a constant vector of the same dimension as Z_k, and the b_{ijk} are known real numbers. The functions $g_k(Z_k)$ of the above form are the *polyhedral convex functions* (PC functions) studied extensively by Rockafellar (1970). Since the maximum or sum of two PC functions is a PC function (Rockafellar, Section 19) it follows that f is a PC function.

Our particular interest is the case where $Z_k \geq 0$ and each component function of $g_k(Z_k)$, $k = 0, 1, \ldots, s$ is an isotone function. We will call this class of functions f the *isotone polyhedral convex* (IPC) functions. A sufficient condition for a PC function to be an IPC

function is that every vector a_{ijk} has all nonnegative entries. In particular, for the special case when f is a linear function with nonnegative coefficients, f is an IPC function. If each function f_k of **PM** is an IPC function we shall see that **PM** is equivalent to a linear programming problem.

Note that $f(D(X))$ is a convex function, in the sense of Dearing, Francis and Lowe, if f is an IPC function.

Informally, we can consider an IPC function f to be one with each entry in each of its component functions a real-valued function of r real variables, expressed using a finite number of max and sum operators, and only these operators. Each operator can be *weighted* in the sense that each element operated on can be multiplied by a nonnegative number. Many location problems are of a "minimax" or "minisum" nature, so that our subsequent results will be applicable to many location problems. When each f function in **PM** is an IPC function, we denote problem **PM** by **PIPC**. **PIPC** does *not* include, as a special case, the n-median or n-center problem for $n \geq 2$, since these problems involve the use of the min-operator, due to the assumption of service of each customer by a nearest facility.

Our intent in studying **PIPC** is to convert it to an equivalent linear program. The way we treat max-operators is well known in the context of converting minimax problems into equivalent constrained problems but, so far as we know, our treatment of *general* IPC functions in a network location context is new.

We caution that our approach for obtaining equivalent linear programs may not be best for specific problem structures obtainable as special cases. Particularly for minimax problems, our introduction of a z variable corresponding to every tree distance may be an extravagant use of variables. For example, the approach we develop, as applied to the 1-center problem, yields an equivalent linear program with $m + 1$ variables and $m + m(m-1)/2$ constraints, whereas the approach of Dearing or Francis yields a linear program with 1 variable and $m(m-1)/2$ constraints; all three approaches use the separation conditions, but the latter two approaches better exploit the minimax structure of the problem. Finding "best" equivalent formulations for the problems studied, in terms of computational effort, is a topic for future research.

In the Appendix, we demonstrate how to transform **PIPC** into an equivalent linear program. We state this linear program, denoted by **PMinLP**, as follows.

Problem PMinLP

$$\text{Minimize} \quad c^T V$$

$$\text{subject to} \quad QZ - PV \leq \beta,$$
$$FZ \leq b,$$
$$AZ \geq d,$$
$$Z, \quad V \geq 0.$$

We will briefly describe the structure of **PMinLP**; more detail appears in the Appendix. Recall that **PIPC** is the special case of **PM** when the functions $f_0, f_1, \ldots, f_p$ are IPC functions. In addition, let us suppose that $f_{r+1} \ldots, f_p$ are linear functions. To represent the linear constraint functions in matrix form, let F be a matrix with rows $f_{r+1}, \ldots, f_p$, and b be a vector with entries $b_{r+1}, \ldots, b_p$. Since the functions $f_0, \ldots, f_r$ are IPC, the vectors c and β, and the matrices Q and P, reflect the composition ideas described earlier in this section. As we show in the Appendix, Q and P are block-diagonal, with each block in P having a staircase structure. Furthermore, each entry in c, Q and F is nonnegative. Finally, $AZ \geq d$ represents the separation conditions. *Our principal conclusion of this section is that* **PIPC** *and* **PMinLP** *are equivalent.* In the next section, we explore solving **PMinLP** by solving its dual.

Certainly if a direct solution approach, such as the primal simplex method, is to be used, we recommend the use of $\alpha Z \geq \delta$ in place of $AZ \geq d$ in **PMinLP**. However, for purposes of exposition, we retain the above formulation. We reserve computational experience with the $\alpha Z \geq \delta$ formulation for future research.

3. Algorithmic Considerations

We develop a means of solving **PMinLP**, as well as the version of **PMinLP** we obtain by replacing A and d by α and δ, respectively, by solving the corresponding dual problem. Our second solution approach is built upon the first, and shares with it common ideas; also, our computational experience is with the first approach.

The dual of **PMinLP**, denoted by **PMaxLP**, is as follows.

Problem PMaxLP

$$\text{Maximize} \quad -\beta^T Y_1 - b^T Y_2 + d^T Y_3$$

$$\text{subject to} \quad -Q^T Y_1 - F^T Y_2 + A^T Y_3 + S_1 = 0,$$
$$P^T Y_1 + S_2 = c,$$
$$Y_1, Y_2, Y_3, S_1, \quad S_2 \geq 0.$$

Let us explore the application of the revised simplex method (Dantzig 1963) to this problem. Note that the dimension of the basis matrix is now q (q is the number

of arcs in the network $N(Z)$) plus the number of rows in the matrix P^T. Observing that **PMaxLP** has an initial basic feasible solution with an identity matrix as a basis matrix, let us suppose we have some given arbitrary basic feasible solution and the corresponding basis inverse matrix, with (Z, V) the corresponding vector of simplex multipliers. Note that Z and V are the vectors of reduced costs for S_1 and S_2, respectively. We list below the column vectors of reduced costs corresponding to Y_1, Y_2, and Y_3:

Y_1: $\beta - QZ + PV$;

Y_2: $b - FZ$; Y_3: $-d + AZ$.

It should be clear that the critical issue is to determine how to do pricing for the columns of A^T, each of which corresponds to a direct path of $N(Z)$. We employ a recommendation of Ford and Fulkerson (1958) who encountered a multicommodity flow problem with a similar mathematical structure.

Note that nonnegativity of the reduced costs for Y_3 is equivalent to the separation conditions. If we apply Dijkstra's (1959) algorithm to $N(Z)$, in the worst case, we may need to compute a shortest direct path from each E node to every other E node, giving an $0(mn(m + n))$ effort for pricing. This order is derived as follows: for every E_i, let $N_i(Z)$ be the subgraph of $N(Z)$ which includes all N nodes and the node E_i. Applying Dijkstra's algorithm to $N_i(Z)$, we find the shortest path from E_i to every N node in an $0(n^2)$ effort, since $N_i(Z)$ has $n + 1$ nodes. Repeating the above for every E node gives an $0(mn^2)$ effort to find the shortest path from every E node to every N node. But then, to find a shortest direct path between any of the $0(m^2)$ pairs of E nodes, we must consider every N node to "match up" the shortest paths, giving $0(m^2n)$ additional effort. This approach allows the pricing of *all* columns of A^T, whereas SLP only prices out all columns when no violated path exists.

While it does not allow the pricing of all columns of A^T (unless no violated path exists), SLP, which works directly with the tree T, can either find a violated path or conclude that none exists in a worst case effort of $0(mn + n^2)$ (achieved when an optimal solution is found). Note that this effort is *linear* in m, as opposed to a quadratic effort (in m) for the other approach; furthermore, we would expect to have m substantially larger than n due to the nature of the location problems being studied. Hence, there seems to be a good reason to study the use of SLP for pricing columns of A^T. We observe that the use of SLP for column pricing is analogous to common computational strategies, as discussed by Chvátal (1983), of pricing out only a limited number of columns to find one to enter the basis.

SLP discovers that no feasible solution exists by discovering a violated path. At some point SLP places a new facility, say NF_j at location x_j, in the tree T for which there exists some other facility, say F_k, at location y_k in T, with

$$d(x_j, y_k) > z_{jk} \tag{7}$$

that is,

$$d(x_j, y_k) - z_{jk} > 0. \tag{8}$$

Here z_{jk} is the upper bound on the distance between NF_j and F_k, and is the length of the arc (NF_j, F_k) in $N(Z)$ joining nodes NF_j and F_k, so that

$$l(NF_j, F_k: Z) = z_{jk}. \tag{9}$$

In this case, it is known (FLR) that there exist some E nodes, say E_h and E_i (possibly $F_k = E_i$) and paths $P(E_h, NF_j)$ and $P(F_k, E_i)$, where

$$P(E_h, E_i) = P(E_h, NF_j), (NF_j, F_k), P(F_k, E_i) \tag{10}$$

satisfies

$$lP(E_h, E_i: Z) < d(v_h, v_i). \tag{11}$$

Thus a separation condition is violated, so no feasible solution to the distance constraints exists.

The interest in FLR was in constructing a feasible solution to the distance constraints or in discovering that no feasible solution exists. There was no reason at that time to actually *construct* a violated path; it was enough to know of the existence of such a path. However, in our case, we need to be able to construct violated paths in order to find columns to make basic. Hence, we augment SLP by developing a procedure to construct violated paths. The basic idea of the procedure is as follows. Each time the location of some new facility, say NF_j, causes a distance constraint to be tight, a record is made of the other facility, say F_i, defining the tight distance constraint. (When more than one distance constraint is tight, we arbitrarily choose one of the F_i which are involved in a tight constraint.) We call NF_j and F_i *stopped* and *stopping facilities*, respectively. In effect, NF_j is labeled from F_i.

When no feasible solution exists to the distance constraints, it is known that all locations made by SLP cause tight distance constraints, and hence cause labeling to occur. Suppose SLP places NF_j at x_j, and some facility F_k at location y_k causes (7) to be satisfied. In this case we have a violated path, and no feasible

solution exists. We can then use the labeling information to trace a sequence of stopping facilities from NF_j until we find a stopping facility which is an existing facility, say E_h; the sequence of facilities defines a path, say $P(E_h, NF_j)$, in $N(Z)$. Likewise, we use the labels to construct a path $P(F_k, E_i)$ in $N(Z)$. Consider the path P defined by (10). Essentially the same approach as in Lemma 5.2 of FLR guarantees

$$lP(E_h, NF_j: Z) = d(v_h, x_j) \tag{12}$$

$$lP(F_k, E_i: Z) = d(y_k, v_i). \tag{13}$$

On adding (9), (12), and (13) we get

$$lP(E_h, E_i: Z) = d(v_h, x_j) + z_{jk} + d(y_k, v_i). \tag{14}$$

Furthermore, essentially the same approach as in Property 5.2 of FLR guarantees

$$d(v_h, v_i) = d(v_h, x_j) + d(x_j, y_k) + d(y_k, v_i). \tag{15}$$

Subtracting (15) from (14) gives

$$lP(E_h, E_i: Z) - d(v_h, v_i) = z_{jk} - d(x_j, y_k). \tag{16}$$

We call the right side of (16) the *gap* for the constraint $d(x_j, y_k) \leq z_{jk}$. Thus, *the reduced cost of the path P is the gap for the constraint causing P to be a violated path.* Also, (8) and (16) give (11), so we conclude that a path defined by (10) is in fact a violated path.

While it is not a part of the formal SLP algorithm, for computational implementations it may be useful to note that we can also explicitly price out other direct paths containing $P(E_h, NF_j)$ which have negative reduced costs in the following way. We check each distance constraint involving new facility j and the existing facilities, as well as each distance constraint involving new facility j and *previously* located new facilities, for a gap. Each negative gap leads (as above) to a corresponding violated path with a negative reduced cost. Among these violated direct paths we can choose one with a most negative reduced cost to make basic.

Of course, it should be clear that SLP also does *implicit* column pricing because as it begins to locate new facilities it begins to satisfy distance constraints, so that separation conditions begin to be satisfied. Satisfied separation conditions correspond to columns with nonnegative reduced costs. When all separation conditions are satisfied, then all reduced costs for Y_3 are nonnegative.

If necessary, cycling can be avoided by using the conventional means of avoiding cycling (Dantzig) for the revised simplex method.

Assume (1) holds as an equality. If P^T has σ rows, the size of the basis is $q + \sigma$, so transforming the basis inverse takes $0((q + \sigma)^2)$ effort, as compared to $0(q)$ effort to price out columns of A^T using SLP. The effort to price out one column of $(-Q, P)^T$ is $0(q + \sigma)$. Thus, the total effort per iteration depends greatly upon the number of columns of $(-Q, P)^T$ which are priced out. In this regard, it may be useful to exploit the sparsity of this matrix. Because q is quadratic in n, it is important to realize that the effort to transform the basis inverse is a fourth degree polynomial in n. Hence this approach may not be a practical one for large n.

For the problem **PMM**, its dual is to maximize $-b^T Y_2 + d^T Y_3$ subject to $-Y_2 + A^T Y_3 + S_1 = c$, Y_2, Y_3, $S_1 \geq 0$. In this case, the effort to price out columns corresponding to entries in Y_3 is just q, so (using SLP) the effort to do all column pricing is order q, and it is the effort to transform the inverse of the basis which is the predominant effort per iteration.

While our computational experience has not kept pace with our algorithm development, we do have some computational experience for **PMM**. We have developed codes that solve **PMM** using the Floyd-Warshall algorithm (Floyd 1962, Warshall 1962), a rudimentary implementation of Dijkstra's algorithm of $0(m(m + n)^2)$, and SLP. Our experience can be summarized as follows. Column pricing with SLP was the most successful of the three approaches we tried. SLP was comparable, in terms of the number of iterations, to complete column pricing using the Floyd-Warshall algorithm, and about ten times faster than either the Dijkstra or the Floyd-Warshall algorithm for column pricing. Regression experience with SLP indicates that the number of iterations grows at a rate that is less than quadratic in basis size.

For the largest problems we solved, of size $(n, m) = (10, 50)$, the average sparsity of the basis inverse exceeded 99%. This sparsity makes the problems we solved behave like much smaller problems in terms of memory requirements and run time. We exploited this sparsity to develop a Pascal code which runs on PC-compatible machines. The average times this code took to solve problems of size (8, 50), (9, 50) and (10, 50) were 37.3, 70, and 450 minutes, respectively; each time is based on a sample size of 10 randomly generated problems, and includes the time needed to maintain a frequency distribution of basis inverse values, as well as to make periodic corrections of the basis inverse.

To summarize, our computational experience

indicates that column generation with SLP is an adequate means of solving **PMM** for the problems we considered. More information on computational testing is available upon request.

We now consider solving problem **PMinLP** in a way that exploits the theorem of Section 1. For simplicity, we denote **PMinLP** by **P**, and **PMaxLP** by **D**. We denote by **P′** the problem obtained by replacing $AZ \geq d$ in **P** by $\alpha\ Z \geq \delta$. Likewise, we denote by **D′** the problem obtained by replacing A and d in **D** by α and δ, respectively. Of course, **D′** is the dual of **P′**. We know, by the theorem of Section 1 and its corollary, that **P** and **P′** have the same optimal solutions and objective function values.

We can solve **P′** by solving **D′** in essentially the same way we solve **P** by solving **D**, except by using an algorithm SLPT in place of the algorithm SLP. SLPT uses SLP to seek a violated path, and then checks for violated triangle inequalities using the nodes in the violated path. The algorithm SLPT works as follows.

Use SLP to check $D(X) \leq Z$. Either SLP constructs X such that $D(X) \leq Z$, or else finds a violated path, say $P = (E_h, N_{[1]}, \ldots, N_{[p]}, E_i)$ in $N(Z)$ of length $lP(E_h, E_i: Z)$ such that

$$lP(E_h, E_i: Z) < d(v_h, v_i). \tag{17}$$

i. If there is a single N node in **P**, (17) defines a violated path constraint in $\alpha\ Z \geq \delta$ and we stop.

ii. If there are exactly two N nodes in P, then (17) becomes

$$z(E_h, N_{[1]}) + z(N_{[1]}, N_{[2]}) + z(N_{[2]}, E_i) < d(v_h, v_i). \tag{18}$$

If $[1] < [2]$ then we check

$$z(E_i, N_{[1]}) \leq z(E_i, N_{[2]}) + z(N_{[2]}, N_{[1]}) \tag{19}$$

and if (19) holds we conclude

$$z(E_h, N_{[1]}) + z(N_{[1]}, E_i) < d(v_h, v_i). \tag{20}$$

Equation 20 defines a violated path constraint in $\alpha\ Z \geq \delta$ and we stop; if (19) fails, then we have a violated triangle inequality and we stop. In case $[2] < [1]$ we check

$$z(E_h, N_{[2]}) \leq z(E_h, N_{[1]}) + z(N_{[1]}, N_{[2]}) \tag{21}$$

and if (21) holds we conclude that

$$z(E_h, N_{[2]}) + z(N_{[2]}, E_i) < d(v_h, v_i). \tag{22}$$

Equation 22 now defines a violated path constraint in $\alpha\ Z \geq \delta$ and we stop; if (21) fails, then we have a violated triangle inequality and stop.

iii. In case there are at least three N nodes in P, then we check the following $p - 2$ ($\leq n - 2$) triangle inequalities

$$z(N_{[1]}, N_{[j]}) \leq z(N_{[1]}, N_{[j-1]}) + z(N_{[j-1]}, N_{[j]}) \quad \text{for } j = 3, \ldots, p. \tag{23}$$

(The choice of triangle inequalities to check is not the only choice.) If at least one of the type-NN inequalities in (23) fails, then we have identified a violated constraint in $\alpha\ Z \geq \delta$ and stop. Otherwise, we use (23) with (17) to conclude

$$z(E_h, N_{[1]}) + z(N_{[1]}, N_{[p]}) + z(N_{[p]}, E_i) < d(v_h, v_i).$$

We proceed as in ii above, checking a type-EN inequality; if it fails we have a violated triangle inequality in $\alpha\ Z \geq \delta$ and stop; otherwise we conclude $z(E_h, N_k) + z(N_k, E_i) < d(v_h, v_i)$ for $k = [1]$ or $k = [p]$ and proceed as in i above.

SLPT outputs either the X it finds such that $D(X) \leq Z$ or else the violated inequality in $\alpha\ Z \geq \delta$ it identifies. As concerns the order of SLPT, it takes $0(n(m + n))$ effort with SLP to find a violated path, and then it takes $0(n)$ effort to find a violated inequality in $\alpha\ Z \geq \delta$. Thus, in $0(n(m + n) + n) = 0(n(m + n))$ effort, SLPT either constructs X such that $D(X) \leq Z$ or finds at least one violated inequality in $\alpha\ Z \geq \delta$.

Let us now relate the above to solving **P′**. We solve **P′** by solving **D′** in the same manner that we solve **P** by solving **D**, except that SLPT is used for column generation with **D′** in the exactly analogous manner that SLP is used for column generation with **D**. Given $Z \geq 0$, we use SLPT for pricing columns of α^T, either identifying a column of α^T to make basic, or else concluding there exists $X \in T^n$ such that $D(X) \leq Z$. Proceeding in this way, at some iteration (assuming no cycling) we conclude that all reduced costs, disregarding those associated with the columns of α^T, are nonnegative, and that there exists $X \in T^n$ such that $D(X) \leq Z$; in this case, we have the following conclusion.

Lemma. *Suppose we have a basic feasible solution to* **D′** *with all reduced costs nonnegative corresponding to the variables* Y_1, Y_2, S_1, *and* S_2 *of* **D′**. *Let* (Z, V) *denote the vector of simplex multipliers for this basic feasible solution.*

If there exists $X \in T^n$ *for which* $D(X) \leq Z$, *then with* $Z^* \equiv D(X)$, (Z^*, V) *is an optimal solution to* **P′**. *Thus* X *is an optimal solution to* **PIPC**.

Proof. Let θ denote the objective function value of the basic feasible solution to **D′**; the formula that

expresses θ in terms of the basis cost vector, the basis inverse, and the right side of the constraints of **D'**, together with the definition of the multipliers, implies

$$\theta = (0^T, c^T)\begin{pmatrix} Z \\ V \end{pmatrix} = c^T V.$$

Our assumptions about the nonnegativity of the reduced costs give

$$Z \geq 0, \quad V \geq 0, \quad QZ - PV \leq \beta, \quad FZ \leq b.$$

Next we observe that each entry in the Q matrix and in the F matrix of **P'** is nonnegative. Since $0 \leq Z^* \leq Z$, it follows that

$$QZ^* - PV \leq QZ - PV \leq \beta, \quad FZ^* \leq FZ \leq b.$$

Also since $Z^* = \mathrm{D}(X)$ and $Z^* \leq Z$, and since tree distances obey the triangle inequalities, we conclude that $\alpha\, Z^* \geq \delta$. Of course Z^*, $V \geq 0$ also, and thus we conclude that (Z^*, V) is a feasible solution to **P'**. But the objective function value of (Z^*, V) is just $c^T V$, and $c^T V = \theta$; thus, the weak duality theorem of linear programming implies (Z^*, V) is an optimal solution to **P'**.

Since **P'** is equivalent to **PIPC**, it follows that X is an optimal solution to **PIPC**.

The attraction of SLPT is that it avoids checking explicitly all the triangle inequality conditions, instead checking at most $0(n)$ of them, and *substituting* the conclusion that there exists X such that $\mathrm{D}(X) \leq Z$ for the conclusion that $\alpha\, Z \geq \delta$. Thus, we get a means of solving the problem which exploits the theorem of Section 1.

To summarize, it is pricing the columns of A^T and α^T that is not done in the conventional way. To do this pricing we check the separation conditions. In order to check the separation conditions we need $Z \geq 0$, which can be assured by continuing to make corresponding slack variables basic whenever an entry in Z becomes negative.

4. An Example

Consider the following example of the multimedian problem, **PMM**. Since our computational experience is for the version of **PMM** with $AZ \geq d$, we use $AZ \geq d$ in our example also.

Let T be a tree with vertices v_1, v_2, v_3 and v_4, and arcs $[v_1, v_4]$, $[v_2, v_4]$, and $[v_3, v_4]$, each of unit length. We wish to locate three new facilities, at locations denoted by (x_1, x_2, x_3), to be determined. For $i = 1, 2, 3, 4$ let E_i correspond to the existing facility i, and for $j = 1, 2, 3$ let N_j represent new facility j.

Let all 15 components of the distance bounds vector b be equal to 2 with the exception of the entries $b(\mathrm{N}_1, \mathrm{N}_2)$, $b(\mathrm{N}_1, \mathrm{N}_3)$ and $b(\mathrm{N}_2, \mathrm{N}_3)$, which are each set to 1. Also let every component of the cost vector c be 0, with the exception of the entries $c(\mathrm{E}_1, \mathrm{N}_1)$, $c(\mathrm{E}_2, \mathrm{N}_2)$ and $c(\mathrm{E}_3, \mathrm{N}_3)$, which are each equal to 1.

We shall see that the (unique) solution to this example is to select x_j, $j = 1, 2, 3$ to be the midpoint of the (unit length) arc $[v_j, v_4]$ of the tree.

We use the above example to illustrate some of the general results and algorithms presented above. First note that the network $\mathrm{N}(Z)$ has a total of 7 nodes and 15 arcs, since $m = 4$ and $n = 3$. Figure 1 shows a sketch of $\mathrm{N}(Z)$. The total number of direct paths connecting pairs of E nodes is 90. Thus, in the formulation

$$\text{minimize} \quad c^T Z$$

$$\text{subject to} \quad AZ \geq d, \quad 0 \leq Z \leq b$$

the separation constraints, $AZ \geq d$, involve 15 variables and 90 constraints. The compact formulation with $AZ \geq d$ replaced by $\alpha\, Z \geq \delta$, based on the theorem, has the same set of variables but only 33 constraints.

We now illustrate the column pricing procedure of Section 3, used to solve the dual of **PMM**, namely

$$\text{maximize} \quad -b^T Y_2 + d^T Y_3$$

$$\text{subject to} \quad -Y_2 + A^T Y_3 + S_1 = c,$$

$$Y_2, Y_3, S_1 \geq 0.$$

The dimension of a (dual) basis is 15. If we start with the basis corresponding to the slacks S_1, the vector of simplex multipliers, Z, is identically zero. Thus the separation conditions $AZ - d \geq 0$ (i.e., the nonnegativity requirement of the reduced costs for Y_3) are all violated. Therefore, each path of $\mathrm{N}(Z)$ is a violated path.

Recall that columns of A^T correspond to direct paths in $\mathrm{N}(Z)$ between pairs of E nodes. Likewise, rows of A^T, and thus the dual equality constraints, correspond to arcs of $\mathrm{N}(Z)$. To demonstrate the pricing mechanism on a nontrivial basis, consider the feasible (dual) basis B defined by the column of A^T associated with the single component (path) of Y_3 connecting E_1 and E_2 via N_1 only. This column, which we take to be the first column of B, is augmented to a full basis as follows. Let the first four rows of B correspond

respectively to the following arcs of N(Z): (E_1, N_1), (E_2, N_2), (E_3, N_3), and (E_2, N_1). Choose as columns 2 and 3 of B the columns associated with components of S_1 corresponding to arcs (E_2, N_2) and (E_3, N_3), respectively. Let the fourth column be associated with the component of Y_2 corresponding to (E_2, N_1). Finally, let the last eleven columns be associated with the components of Y_2 corresponding to the arcs not represented above. Permute columns (and rows) 5 through 15 so that column i is $-e_i$, $i = 5, \ldots, 15$. Therefore, the basis matrix appears as

$$B = \left[\begin{array}{ccc|cccccccccc} 1 & & & & & & & & & & & \\ & 1 & & & & & & & & & & \\ & & 1 & & & & & & & & & \\ 1 & & & -1 & & & & & & & & \\ \hline & & & & -1 & & & & & & & \\ & & & & & \ddots & & & & & & \\ & & & & & & \ddots & & & & & \\ & & & & & & & -1 & & & & \end{array}\right].$$

The basis matrix is equal to its inverse. Using the appropriate objective coefficients, it is easily verified that Z, the vector of multipliers associated with this basis, is given by $z(E_1, N_1) = z(E_2, N_2) = z(E_3, N_3) = 0$, $z(N_1, N_2) = z(N_1, N_3) = z(N_2, N_3) = 1$; the other nine components are all equal to 2.

To find a violated path (equivalently, to find the reduced costs for Y_3) we test the separation conditions with SLP, which works directly on the original tree. Note that the input to SLP consists of the above vector Z, which expresses the upper bounds on the distances between the facilities. The separation conditions are not satisfied. In particular, $z(E_1, N_1) = z(E_2, N_2) = 0$ imply $x_1 = v_1$ and $x_2 = v_2$. But then $d(x_1, x_2) = d(v_1, v_2) = 2$, contradicting $z(N_1, N_2) = 1$. Thus, (E_1, N_1, N_2, E_2) is a (direct) violated path of N(Z). This path has length $0 + 1 + 0 < 2 = d(v_1, v_2)$, and a gap of $2 - 1 = 1$.

This first basis example also applies for **PMM** with $\alpha Z \geqslant \delta$, since the only column of A^T in the basis corresponds to a column with a single N node.

As another illustration of the pricing procedure, consider the following (optimal) dual basis for the above example. Let rows 1, 2 and 3 correspond respectively to the arcs (E_1, N_1), (E_2, N_2), and (E_3, N_3). There are three columns of A^T in the basis; columns 1–3 correspond to the following paths on N(Z): P_1 = (E_2, N_2, N_3, E_3); P_2 = (E_1, N_1, N_3, E_3); P_3 = (E_1, N_1, N_2, E_2). Let columns 4–15 be associated with the components of Y_2 corresponding to all arcs of N(Z) but (E_i, N_i), $i = 1, 2, 3$. Of these, columns (and rows) 4, 5 and 6 correspond to (N_2, N_3), (N_1, N_3) and (N_1, N_2), respectively. Permute columns (and rows) 7–15 so that column i is $-e_i$, $i = 7, \ldots, 15$. Thus the basis matrix is given by

$$B = \left[\begin{array}{ccc|cccccccccc} 0 & 1 & 1 & & & & & & & & & \\ 1 & 0 & 1 & & & & & & & & & \\ 1 & 1 & 0 & & & & & & & & & \\ \hline 1 & & & -1 & & & & & & & & \\ & 1 & & & -1 & & & & & & & \\ & & 1 & & & -1 & & & & & & \\ \hline & & & & & & \ddots & & & & & \\ & & & & & & & \ddots & & & & \\ & & & & & & & & -1 & & & \end{array}\right],$$

It can be verified that the vector of multipliers, Z, associated with this basis is given by $z(E_1, N_1) = z(E_2, N_2) = z(E_3, N_3) = 1/2$, $z(N_1, N_2) = z(N_1, N_3) = z(N_2, N_3) = 1$, and the other nine components are all equal to 2. Applying SLP with the above bounds indicates that there are no violated paths, i.e., the separation conditions are satisfied, and thus all reduced costs are nonnegative. The *only* feasible solution for the separation conditions with the above bounds is to select x_j, the location of new facility j, $j = 1, 2, 3$, as the midpoint of the arc $[v_j, v_4]$ of the tree. Consequently, we get a unique optimal solution to our example problem.

Appendix

We demonstrate here how to transform **PIPC** into an equivalent linear program. With f an IPC function defined by $f(Z) = g_s(\cdots(g_1(g_0(Z)))\cdots)$, repeated use of the Monotonicity Lemma implies the inequality $f(D(X)) \leqslant b$ is equivalent to $D(X)$, $\leqslant Z$, and

$$\begin{aligned} g_0(Z) - R_1 &\leqslant 0, \\ g_1(R_1) - R_2 &\leqslant 0, \\ &\cdots \\ g_{s-1}(R_{s-1}) - R_s &\leqslant 0, \\ g_s(R_s) &\leqslant b. \end{aligned} \tag{24}$$

$$Z \geqslant 0; \quad R_k \geqslant 0, \quad k = 1, \ldots, s. \tag{25}$$

It should be clear that there exist matrices $M_0, \ldots, M_s$; $N_1, \ldots, N_s$; vectors $\mu_1, \ldots, \mu_s$; and a vector b^+,

so that the inequalities (24) are also equivalent to the linear inequalities

$$\begin{aligned} M_0 Z - N_1 R_1 &\leq \mu_1, \\ M_1 R_1 - N_2 R_2 &\leq \mu_2, \\ &\cdots \\ M_{s-1} R_{s-1} - N_s R_s &\leq \mu_s, \\ M_s R_s &\leq b^+. \end{aligned} \tag{26}$$

For example, M_0 has a row for each term inside every max-operator in $g_0(Z)$ as does μ_1; entries in μ_1 are the negatives of the additive constants inside the max-operators. N_1 has rows that are unit vectors, with unit vectors repeated for each term in every max-operator in $g_0(Z)$. Entries in b^+ repeat corresponding entries of b for each term in every max-operator in $g_s(R_s)$. Note that all the M matrices are nonnegative, and the N matrices have entries of 0 and 1. Furthermore, the M and N matrices can have more rows than columns due to the occurrence of a max-operator.

Equivalent to (26), we also have matrices P^* and Q^*, and vectors V and β, giving (27) as

$$Q^* Z - P^* V \leq \beta \tag{27}$$

where V is the column vector with (partitioned) entries R_1 through R_s

$$P^* = \begin{bmatrix} N_1 & 0 & & 0 \\ -M_1 & N_2 & & 0 \\ 0 & -M_2 & & 0 \\ & & \cdots & \\ 0 & 0 & & N_s \\ 0 & 0 & & -M_s \end{bmatrix}$$

$$Q^* = \begin{bmatrix} M_0 \\ 0 \\ 0 \\ \cdots \\ 0 \\ 0 \end{bmatrix} \qquad \beta = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \cdots \\ \mu_s \\ b^+ \end{bmatrix}.$$

Consider **PIPC** when the objective function, and the first r constraint functions, are IPC functions. Let the remaining constraint functions of **PIPC** be linear functions, i.e., $f_i^T Z \leq b_i$, $i = r+1, \ldots, p$. Using the results above, we can write our problem as the following, denoted by $\mathbf{PMinLP}_0$, where c_0 is a nonnegative vector.

Problem $\mathbf{PMinLP}_0$

Minimize $c_0^T V_0$

subject to

$$Q_i Z - P_i V_i \leq \beta_i, \quad i = 0, 1, \ldots, r,$$

$$f_i^T Z \leq b_i, \quad i = r+1, \ldots, p,$$

$$\mathrm{D}(X) \leq Z,$$

$$Z \geq 0; \quad V_i \geq 0, \quad i = 0, 1, \ldots, r.$$

Making use of the path constraints, we write $\mathbf{PMinLP}_0$ equivalently as **PMinLP**, given at the end of Section 2.

Acknowledgment

This research was supported in part by the National Science Foundation, grants ECS-8317026, ECS-8317138, ECS-8612907, and ECS-8612911.

References

BLAND, R. G., D. GOLDFARB AND M. J. TODD. 1981. The Ellipsoid Method: A Survey. *Opns. Res.* **29,** 1039–1091.

CHAN, A. W., AND R. L. FRANCIS. 1976. A Round-Trip Location Problem on a Tree Graph. *Trans. Sci.* **10,** 35–51.

CHVÁTAL, V. 1983. *Linear Programming.* W. H. Freeman, New York.

DANTZIG, G. 1963. *Linear Programming and Extensions.* Princeton University Press, Princeton, N.J.

DEARING, P. M. 1977. Minimax Location Problems with Nonlinear Costs. *J. Res. Nat. Bur. Standards* **28,** 65–72.

DEARING, P. M., AND R. L. FRANCIS. 1974a. A Network Flow Solution to a Multifacility Minimax Location Problem Involving Rectilinear Distances. *Trans. Sci.* **8,** 126–141.

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

DEARING, P. M., AND J. J. LANGFORD. 1975. The Multifacility Total Cost Location Problem on a Tree Network. Technical Report No. 209, Department of Mathematical Sciences, Clemson University, Clemson, S.C.

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. *Numer. Math.* **1,** 269–271.

ERKUT, E., R. L. FRANCIS AND T. J. LOWE. 1988. A Multimedian Problem with Interdistance

Constraints. *Environ. Plan. B: Plann. Design* **15,** 181–190.

FLOYD, R. W. 1962. Algorithm 97, Shortest Path. *Commun. ACM* **5,** 345–345.

FORD, L. R., AND D. R. FULKERSON. 1958. A Suggested Computation for Maximal Multi-Commodity Network Flows. *Mgmt. Sci.* **5,** 97–101.

FRANCIS, R. L. 1977. A Note on a Nonlinear Minimax Location Problem in a Tree Network, *J. Res. Nat. Bur. Standards* **28,** 73–80.

FRANCIS, R. L., AND P. B. MIRCHANDANI (Eds.) 1989. *Discrete Location Theory.* John Wiley & Sons, New York.

FRANCIS, R. L., AND J. A. WHITE. 1974. *Facility Layout and Location: An Analytical Approach.* Prentice-Hall, Englewood Cliffs, N.J.

FRANCIS, R. L., T. J. LOWE AND H. D. RATLIFF. 1978. Distance Constraints for Tree Network Multifacility Location Problems. *Opns. Res.* **26,** 570–596.

GRÖTSCHEL, M., L. LOVÁSZ AND A. SCHRIJVER. 1981. The Ellipsoid Method and its Consequences in Combinatorial Optimization. *Combinatorica* **1,** 169–197.

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. 1985. Medi-Centers of a Tree. *Trans. Sci.* **19,** 246–260.

HSU, W. L., AND G. L. NEMHAUSER. 1979. Easy and Hard Bottleneck Location Problems, *Discrete Appl. Math.* **2,** 77–91.

KARIV, O., AND S. L. HAKIMI. 1979a. An Algorithmic Approach to Network Location Problems. Part 1: The *p*-Centers. *SIAM J. Appl. Math.* **37,** 513–538.

KARIV, O., AND S. L. HAKIMI. 1979b. An Algorithmic Approach to Network Location Problems. Part 2: The *p*-Medians. *SIAM J. Appl. Math.* **37,** 539–560.

KARMARKAR, N. 1984. A New Polynomial-Time Algorithm for Linear Programming. *Combinatorica* **4,** 373–396.

KOLEN, A. 1981. Equivalence Between the Direct Search Approach and the Cut Approach to the Rectilinear Distance Location Problem. *Opns. Res.* **29,** 616–620.

KOLEN, A. 1982. *Location Problems on Trees and in the Rectilinear Plane.* Stitchting Mathematisch Centrum, Kruislaan 413, 1098 SJ, Amsterdam, The Netherlands.

KOLEN, A., AND A. TAMIR. 1989. Covering Problems, a chapter in *Discrete Location Theory,* R. L. Francis and P. B. Mirchandani (eds.). John Wiley & Sons, New York.

LOVÁSZ, L. 1986. An Algorithmic Theory of Numbers, Graphs, and Convexity. *SIAM Monogr.* Philadelphia.

MOON, D., AND S. S. CHAUDHRY. 1984. An Analysis of Network Location Problems with Distance Constraints. *Mgmt. Sci.* **30,** 290–307.

MORE, J. J., AND W. C. REINBOLDT. 1973. On *P* and *S* Functions and Related Classes of *N*-Dimensional Nonlinear Mappings, *Linear Algebra Appl.* **6,** 45–68.

PICARD, J-C., AND H. D. RATLIFF. 1978. A Cut Approach to the Rectilinear Distance Facility Location Problem. *Opns. Res.* **28,** 422–433.

REINBOLDT, W. C. 1970. On *M*-Functions and Their Applications to Gauss-Seidel Iterations and to Network Flows. *J. Math. Anal. Appl.* **32,** 274–307.

ROCKAFELLAR, R. T. 1970. *Convex Analysis.* Princeton University Press, Princeton, N.J.

TANSEL, B. C., R. L. FRANCIS AND T. J. LOWE. 1980. Binding Inequalities for Tree Network Location Problems with Distance Constraints. *Trans. Sci.* **14,** 107–124.

TANSEL, B. C., R. L. FRANCIS AND T. J. LOWE. 1983. Location on Networks: A Survey. *Mgmt. Sci.* **29,** 482–511.

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,** 725–744.

WARSHALL, S. 1962. A Theorem on Boolean Matrices. *J. Assoc. Comput. Mach.* **9,** 11–12.