J. Opl Res. Soc. Vol. 35, No. 8, pp. 741–748, 1984
Printed in Great Britain.
0160-5682/84 \$3.00 + 0.00

The p-Centre Problem—Heuristic and Optimal Algorithms

ZVI DREZNER
School of Management, The University of Michigan–Dearborn

The p-centre problem, or minimax location–allocation problem in location theory terminology, is the following: given n demand points on the plane and a weight associated with each demand point, find p new facilities on the plane that minimize the maximum weighted Euclidean distance between each demand point and its closest new facility. We present two heuristics and an optimal algorithm that solves the problem for a given p in time polynomial in n. Computational results are presented.

INTRODUCTION

THERE ARE two equivalent approaches to formulating and solving the p-centre problem. The first calls for the location of p new facilities that minimize the maximal weighted distance between each demand point and its closest new facility. The second approach consists of two phases: first, the set of demand points is partitioned into p disjoint subsets, and second, the best location of a new facility for each subset is found. This location must minimize the maximal weighted distance between the new facility and the demand points of this subset. The maximum among these weighted distances for all subsets must be minimized by the choice of the subsets.

Many real life problems can be modelled after this formulation. In fact, almost every minimax location problem with one new facility turns to a p-centre problem when several new facilities have to be located. For instance, the problem of locating a fire station that will serve some demand points in the shortest possible time is a single facility minimax location problem. Location of several fire stations is a p-centre problem. By the first formulation: we must minimize the maximal distance to the closest fire station. By the second formulation: we must find a partition of the set of demand points into p zones and assign a fire station to each zone minimizing the maximal distance between the fire station and the demand points in the zone. The maximal distance in the sytem should be minimized.

Let $X = (x, y)$, $X_i = (x_i, y_i)$. The first formulation is as follows.

Find locations for p new facilities, X_i for $i = 1, \dots, p$, in order to:

$$\underset{X_1,\dots,X_p}{\text{minimize}} \left\{ \max_{1 \leqslant i \leqslant n} \left\{ \min_{1 \leqslant j \leqslant p} \{D_i(X_j)\} \right\} \right\} \tag{1}$$

where:

$D_i(X) = w_i[(x - a_i)^2 + (y - b_i)^2]^{1/2}$;
X_j for $j = 1, \dots, p$ is the location of new facility j;
(a_i, b_i) for $i = 1, \dots, n$ is the location of demand point i;
w_i is the weight associated with demand point i.

The second formulation involves zero–one variables. Let

$$z_{ij} = \begin{cases} 1 & \text{if demand point } i \text{ is assigned to new facility } j; \\ 0 & \text{otherwise.} \end{cases}$$

The problem can now be stated as

$$\underset{X_1,\dots,X_p}{\text{minimize}} \left\{ \max_{\substack{1 \leqslant i \leqslant n \\ 1 \leqslant j \leqslant p}} \{z_{ij} D_i(X_j)\} \right\} \tag{2}$$

subject to:

$$\sum_{j=1}^{p} z_{ij} = 1 \quad \text{for} \quad i = 1, \ldots, n; z_{ij} \in \{0, 1\}.$$

HEURISTICS

Two heuristics that may be used for large problems and may also shorten the run time of the optimal algorithm are presented in this section. These heuristics resemble algorithms for other location-allocation problems.[1-5]

First let us define some concepts that we shall need later. Let N be the set of indices of demand points: $N = \{1, \ldots, n\}$, and let I be a subset of N. Let $F(I)$ be the optimal value of the objective function for the single facility minimax problem (also referred to as the 1-centre problem) consisting of the demand points of I. $F(I)$ is given by:

$$F(I) = \min_{X} \left\{ \max_{i \in I} \{D_i(X)\} \right\} \tag{3}$$

and let $X^*(I)$ be the optimal point of problem (3). $X^*(I)$ is well defined since there is a unique minimal point of problem (3).[6]

Let $\alpha = \{I_1, \ldots, I_p\}$ be a family of subsets of N. If

$$\bigcup_{k=1}^{p} I_k = N,$$

we call α a complete division of N. If α is a partition of N, we call it a complete disjoint division of N. Let F_α be the value of the object function for the complete division. F_α is given by:

$$F_\alpha = \max_{j} \{F(I_j)\}, \tag{4}$$

where

$$\alpha = \{I_1, \ldots, I_p\}.$$

The following lemma is self evident.

Lemma1:

If $I \subset K$, then $F(I) \leqslant F(J)$.

For a single facility problem based on all demand points of the set I, there exists a subset $B(I)$ of no more than three demand points with the following properties:[6,7]

Property 1: $F(B(I)) = F(I)$
Property 2: $X^*(B(I)) = X^*(I)$. $B(I)$ will be referred to as the 'binding subset'.

First heuristics (H-1)

Choose p starting points $X_1^{[0]}, \ldots, X_p^{[0]}$. We suggest choosing p points out of the n demand points. Let k be the iteration number and suppose that $X_1^{[k]}, \ldots, X_p^{[k]}$ are given. A set of demand points is assigned to each centre such that each demand point is assigned to the closest centre. In fact, the p centres define p Voronoi polygons (called also Thiessen polygons).[8] The partitioning is done according to these polygons.

Set:

$$I_j^{[k]} = \{i \mid D_i(X_j^{[k]}) \leqslant D_i(X_r^{[k]}) \quad \text{for} \quad r = 1, \ldots, p\}.$$

Now, we recalculate the locations for the centres. The new centre for each set is the solution to the one-centre problem defined by the set:

$$X_j^{[k+1]} = \begin{cases} X^*(I_j^{[k]}) \mid I_j^{[k]} \neq \emptyset \\ X_j^{[k]} \mid I_j^{[k]} = \emptyset \end{cases} \quad \text{for } j = 1, \ldots, p.$$

Stop when $X_j^{[k+1]}$ for $j = 1, \ldots, p$. The solution is the complete disjoint division $\alpha^{[1]} = \{I_1^{[1k]}, \ldots, I_p^{[k]}\}$. Note that if demand point i can belong to more than one set, assign it arbitrarily to one set.

As a second phase heuristic, try rearrangements of one demand point at a time in order to find a better value of the objective function. There are $n(p-1)$ possible rearrangements, but not all of them can *a priori* reduce the value of the objective function. Consider a rearrangement that removes demand point i from I_j and add it to another subset, I_k. By Lemma 1, only the value of $F(I_j)$ may decrease. The value of the objective function, F_α, may decrease only if $F(I_j) = F_\alpha$. When $F(I_j) = F_\alpha$, refer to I_j as an *extremal subset.*

Lemma 2:

If $i \in I$ and $i \notin B(I)$, then $F(U - \{i\}) = F(I)$.

Proof:

By Lemma 1, since $B(I) \subseteq I - \{i\} \subseteq I$, $F(B(I)) \leqslant F(I - \{i\}) \leqslant F(I)$. By property 1, $F(B(I)) = F(I)$. Hence, $F(I - \{i\}) = F(I)$. □

By Lemma 2 the value of $F(I_j)$ may decrease by the rearrangement only if $i \in B(I_j)$. Thus we have to check only rearrangements of demand points that belong to the binding subset of an extremal subset. This is stated explicitly in the following.

Heuristic 2 (H-2)

Let $B' = \{i \mid i \in B(I_j)$ and I_j is an extremal subset$\}$. Select $i \in B'$ and let $i \in I_j$. Check if $F(I_r U\{i\}) < F_\alpha$ for $r \neq j$. If $F(I_r U\{i\}) < F_\alpha$ for some r, then modify the partition as follows: $I_r = I_r U\{i\}$, $I_j = I_j - \{i\}$. Repeat until no further changes are possible.

We propose to apply H-1 and H-2 alternately until neither changes the partition. Heuristics H-1 and H-2 have small computational requirements even for quite large problems. See the computational results section for details.

AN OPTIMAL ALGORITHM

In this section we present a polynomial algorithm for finding optimal solutions for the p-centre problem. The main step in the proposed optimal algorithm is the following: given a value F_0 of the objective function, either find a better solution, or prove that there is no better solution. This step requires the solution of a set-covering problem.[9] The following definitions and theorems are needed for the development of the optimal algorithm.

Definition 1: maximal set

A set of indices, I, $I \subseteq N$, is a maximal set with respect to F_0, if the following conditions hold:

(i) $F(I) < F_0$;
(ii) for every $i \notin I$, $F(IU\{i\}) \geqslant F_0$.

The following two lemmas are obvious.

Lemma 3:

A set J is maximal if and only if $F(J) < F_0$ and for every $I \supset J$ $F(I) \geqslant F_0$.

Lemma 4:

For every set J which satisfies $F(J) < F_0$ there exists a maximal set, I, so that $I \supseteq J$.

Theorem 1:

There exists a solution $\alpha = \{I_1, \ldots, I_p\}$ with $F_\alpha < F_0$ if and only if there exists a complete division into maximal sets.

Proof:

If there exists a solution $\alpha = \{I_1, \ldots, I_p\}$ with $F_\alpha < F_0$, then $F(I_j) < F_0$, for $j = 1, \ldots, p$. Hence, by Lemma 4, for each set I_j there exists a maximal set $I_j^+, I_j^+ \supseteq I_j$. $\alpha^+ = \{I_1^+, \ldots, I_p^+\}$ is a complete division into maximal sets.

On the other hand, there exists a complete division into maximal sets $\alpha = \{I_1, \ldots, I_p\}$. $F(I_j) < F_0$, so $F_\alpha < F_0$. If there are demand points common to several sets, we can choose for each of them one set only, getting a complete disjoint division $\alpha^+ = (I_1^+, \ldots, I_p^+\}$ with $I_j^+ \subseteq I_j$ for $i = 1, \ldots, p$. By Lemma 1, $F(I_j^+) \leqslant F(I_j)$. Since $F(I_j) < F_0, F_{\alpha^+} < F_0$.

Based on Theorem 1, we propose the following optimal algorithm. Find a feasible solution to the p-centre problem using any heuristic. Let the Value of the objective function of this feasible solution be F_0. Find all maximal sets for this F_0 and determine the minimum number of maximal sets required to cover N. If the minimum number of sets is greater than p, there is no feasible solution with value less than F_0. But if the minimum number of sets is less than or equal to p, the solution to the set-covering problem leads to an improved solution to the p-centre problem. We can apply H-1 and H-2 to this improved solution and use the resulting value of the objective function as F_0 for the next iteration of the optimal algorithm.

In order to implement the optimal procedure, we show how to find all maximal sets for a given F_0, and how to solve the associated set-covering problem.

Finding All Maximal Sets

Definition 2: closure of a set

The closure of J, $C(J)$, is:

$$C(J) = \{i \mid D_i(X^*(J)) \leqslant F(J)\}. \tag{5}$$

The following properties are self-evident by the definition of $C(J)$:

Property 3: $J \subseteq C(J)$.
Property 4: $F(C(J)) = F(J)$.

Theorem 2:
For a maximal set I: $I = C(B(I))$.
Proof:
By property 1, $F(B(I)) = F(I)$. By property 2, $X^*(B(I)) = X^*(I)$. Thus $C(B(I)) = C(I)$. By property 3, $C(I) \supseteq I$. Now, if $C(I) \supset I$, then I is not maximal since $F(C(I)) = F(I)$ by property 4. Thus $C(I) = I$ and the theorem follows. □

There are 2^n possible subsets of N. Let N_3 be the set of all subsets of N with no more than three demand points. By Theorem 2, we must check only $n + n(n-1)/2 + n(n-1)(n-2)/6 = n(n^2+5)/6$ sets as possible maximal sets; thus checking all possible sets for maximality is a polynomial algorithm. For all $J \in N_3$, construct $C(J)$. If $F(C(J)) < F_0$, and for all $j \not\in C(J) F(C(J) \cup \{j\}) \geqslant F_0$, then $C(J)$ is a maximal set. Otherwise, $C(J)$ is not maximal.

As a corollary of the theorem in Drezner,[14] it can be easily shown that the number of maximal sets for a given F_0 is bounded by $n(n-1)$.

We propose essentially the same algorithm with some short-cuts. For the short-cuts we present the following lemmas, the proofs of which are trivial.

Lemma 5:
Given $J \in N^3$, if there exists $j \in N$ such that $F(J) < D_j(X^*(J)) < F_0$, then $C(J)$ is not a maximal set.

Lemma 6:
Given a subset J of three elements, if there exists a $j \in J$ such that $j \notin B(J)$, then $C(J - \{j\}) = C(J)$.

The following algorithm finds all maximal sets for a given F_0. Let M be the set of all maximal sets. A maximal set can be defined by a binding set of one, two or three demand points. In Step 2 we find all maximal sets defined by a singleton binding set. In Steps 3–11 all maximal sets defined by a binding set of two demand points are found, and in Steps 12–18 possible binding sets of three demand points are checked. Note that in order to avoid duplication, we check binding sets $\{i, j\}$ and $\{i, j, k\}$ where $i \leqslant j \leqslant k$.

Algorithm for finding maximal sets

Step 1: Set $M = \emptyset$.

Step 2: Add all singleton maximal sets to M by checking all $\{i\}$ for maximality. If $F(\{i,j\}) \geqslant F_0$ for every $j \neq i$, then add $\{i\}$ to M (by definition of a maximal set).
Step 3: Set $i = 1$; $j = 2$.
Step 4: Compute $F(\{i,j\})$, $X^*(\{i,j\})$. If $F(\{i,j\}) \geqslant F_0$, then $\{i,j\}$ cannot generate a maximal set. Go to Step 20.
Step 5: Set $k = 1$; $J = \emptyset$. (J will accumulate all demand points in $C(\{i,j\})$.)
Step 6: If $D_k(X^*(\{i,j\})) \leqslant F(\{i,j\})$, then k is in the closure of $\{i,j\}$. Add k to J and go to Step 8.
Step 7: If $D_k(X^*(\{i,j\})) < F_0$, then $\{i,j\}$ cannot generate a maximal set by Lemma 5. Go to Step 10 to check possible triplets which include $\{i,j\}$.
Step 8: Set $k = k + 1$. If $k \leqslant n$, go to Step 6.
Step 9: If for every $s \notin J$ $F(JU\{s\}) \geqslant F_0$, add J to M since J is maximal by Definition 1.
Step 10: If $j = n$, go to Step 21.
Step 11: Set $k = j + 1$.
Step 12: If $D_k(X^*(\{i,j\})) \leqslant F(\{i,j\})$, then $C(\{i,j,k\}) = C(\{i,j\})$ by Lemma 6. Go to Step 19. Also, if $D_i(X^*(\{j,k\})) \leqslant F(\{j,k\})$ or $D_j(X^*(\{i,k\})) \leqslant F(\{i,k\})$, go to Step 19.
Step 13: Compute $F(\{i,j,k\})$, $X^*(\{i,j,k\})$. If $F(\{i,j,k\}) \geqslant F_0$, then $\{i,j,k\}$ cannot generate a maximal set. Go to Step 19.
Step 14: Set $r = 1$; $J = \emptyset$. (J will accumulate all demand points in $C(\{i,j,k\})$.)
Step 15: If $D_r(X^*(\{i,j,k\})) \leqslant F(\{i,j,k\})$, then r is in $C(\{i,j,k\})$. Add r to J and go to Step 17.
Step 16: If $D_r(X^*(\{i,j,k\})) < F_0$, then $\{i,j,k\}$ cannot generate a maximal set by Lemma 5. Go to Step 19.
Step 17: Set $r = r + 1$. If $r \leqslant n$ go to Step 15.
Step 18: If for every $s \notin J$ $F(JU\{s\}) \geqslant F_0$, add J to M.
Step 19: Set $k = k + 1$. If $k \leqslant n$, go to Step 12.
Step 20: Set $j = j + 1$. If $j \leqslant n$, go to Step 4.
Step 21: Set $i = i + 1$. If $i = n$, stop.
Step 22: Set $j = j + 1$. Go to Step 4.

Solving the set-covering problem

Let $M = \{J_1, \ldots, J_m\}$ be the set of all maximal sets. Let

$$a_{ij} = \begin{Bmatrix} 1 \mid i \in J_j \\ 0 \mid i \notin J_j \end{Bmatrix} \quad \text{for} \quad j = 1, \ldots, m; i = 1, \ldots, n.$$

The problem (for $m > p$) is: find $u_j \in \{0, 1\}$ for $j = 1, \ldots, m$ subject to:

$$\sum_{j=1}^{m} a_{ij} u_j \geqslant 1 \quad \text{for} \quad i = 1, \ldots, n. \tag{6}$$

$$\sum_{j=1}^{m} u_j = p.$$

Problem (6) can be solved by any integer programming method,[10] or by special methods for set-covering problems.[9] The set-covering problem can also be solved by total enumeration, and for a fixed p this approach is polynomial in n. That is because $m \leqslant n(n-1)$, and thus $\binom{m}{p} \leqslant n^{2p}$. The labour involved in checking each possibility is of order n. Therefore, the complexity of the total enumeration approach to the set-covering problem is bounded by $0(n^{2p+1})$.

Each iteration of the proposed optimal algorithm involves polynomial algorithms for finding all maximal sets and solving the set-covering problem. We prove that the number of iterations is bounded by $0(n^3)$, and therefore the algorithm is polynomial.

Theorem 3:
The optimal algorithm solves the problem in polynomial time.
Proof:
We prove that the number of iterations is polynomial and the theorem follows. There are at most $n(n^2+5)/6$ maximal subsets for all possible values of F_0. Therefore, there are at most $n(n^2+5)/6$ possible values of F_0 at the end of each iteration. Since the value of F_0 is decreasing every iteration, there can be $n(n^2+5)/6$ iterations at most. $\square$

One can argue that since we must check subsets created by no more than three demand points, then for $p \leqslant 5$ total enumeration is of lower complexity. Indeed, choose a subset $J_1 \subseteq N$ of up to three points, find $C(J_1)$, $F(J_1)$, then find J_2 of up to three demand points out of $N - C(J_1)$, and so on $p-1$ times, then find $F(N - J_1 - J_2 - \ldots - J_{p-1})$, getting F_α for this disjoint division into subsets. The complexity of the total enumeration is $0(n^{3p-3})f(n)$, where $f(n)$ is the complexity of solving a 1-centre problem of n points. When all weights are equal, then the best known algorithm has a complexity of $0(n \log n)$.[8] Therefore, for the equal weights problem, total enumeration is of complexity bounded by $0(n^{3p-2} \log n)$, which is better than our bound of $0(n^{2p+4})$ for $p \leqslant 5$. $f(n)$ for problems with different weights is probably different. The Elzinga–Hearn[11] algorthim can be extended to unequal weights problems. The complexity of this extended algorithm is $0(n^4)$ but empirically seems to behave like $0(n)$.[6,12] The lowest known complexity is probably $0(n^3)$ by the trivial algorithm of picking the greatest value of the object function (calculated in $0(1)$) among all pairs and triplets of demand points. A solution within a given accuracy of $\varepsilon > 0$ can be found in $0(n \log \varepsilon)$.[13] These are worst case bounds. Practically, as demonstrated by the computational results, the bound of $0(n^{2p+4})$ for the complexity of the optimal algorithm is quite 'generous'. No problem required more than four iterations, even though the bound for a 50 points problem is more than 20,000. The problem of $n = 50$; $p = 3$ was solved by our optimal algorithm in 11.34 seconds, while total enumeration will involve about 1.5×10^{12} triplets and the problem will be solved in 17 days if checking each triplet takes 10^{-6} seconds.

Note that the optimal algorithm can be slightly modified to reduce the bound on the complexity. The method described above of reducing the value of F_0 each iteration can be replaced by a bisection on the values of F_0. Find all $n(n^2+5)/6$ values of F_0 and sort them. This is done only once in the algorithm in a time of $0(n^3 \log n)$. By a bisection on this vector of F_0's, we need only $0(\log n)$ iterations, rather than our bound of $0(n^3)$. The bound for the complexity of the modified algorithm is $0(n^{2p+1} \log n)$. In practice we do not recommend taking this approach, because in our experience the number of iterations in our method is generally lower than the number of iterations in the bisection approach.

A MODIFIED p-CENTRE PROBLEM

The modified 1-centre problem[14] can be extended to a modified p-centre problem. We have developed here the tools needed for the solution of this problem too. Let us define for a given F_0:

$$I(X) = \{i \mid D_i(X_j) \leqslant F_0 \quad \text{for some} \quad 1 \leqslant j \leqslant p\}.$$

The problem is:

$$\underset{X_1, \ldots, X_p}{\text{minimize}} \left\{ \sum_{i \in I(X)} p_i \right\}, \tag{7}$$

where $p_i > 0$ is a coefficient of importance associated with demand point i for $i = 1, \ldots, n$.

The solution to problem (7) can be found as follows. Construct all maximal sets for F_0. Find among all unions of p maximal sets, the union for which Σp_i is maximal. The effort involved in doing this is the same as the effort involved in the solution of problem (6) by total enumeration.

TABLE 1. RUN TIMES (SEC) FOR THE HEURISTIC APPROACH

n	$p = 2$	$p = 3$	$p = 5$	$p = 10$
50	0.01	0.01	0.02	0.03
100	0.01	0.02	0.02	0.03
500	0.04	0.11	0.20	0.59
1000	0.09	0.15	0.29	0.85
1500	0.22	0.22	0.50	1.05
2000	0.29	0.55	1.08	1.27

TABLE 2. COMPARATIVE RESULTS

		Proposed algorithms			Complete enumeration	$\frac{F(X^{H})^*}{F(X^*)}$
p	n	Iter.	Values of m	Time (sec)	time (sec)	
2	10	1	10	0.02	0.02	1.000
2	20	2	16,14	0.17	0.24	1.044
2	30	1	33	0.47	1.40	1.000
2	40	3	33,17,34	1.99	4.31	1.139
2	50	1	75	1.82	9.33	1.000
2	60	3	62,25,63	7.02	20.99	1.106
3	10	2	8,7	0.03	0.05	1.783
3	20	2	30,18	0.25	9.62	1.018
3	30	2	52,32	0.93	†	1.277
3	40	3	41.70.47	3.37	†	1.277
3	50	4	154,75,81,121	11.34	†	1.277
4	10	1	6	0.02	0.06	1.000
4	20	2	30,24	0.26	†	1.033
4	30	2	50,41	1.30	†	1.017
4	40	1	73	4.98	†	1.000
5	10	1	5	0.01	0.06	1.000
5	20	2	29,34	1.43	†	1.085
5	30	3	46,50,58	19.91	†	1.075

*X^{H}—the heuristic solution point.
X^*—the optimal solution point.
†Time is at least 30 seconds.

Since we perform only one iteration of the ordinary optimal algorithm, the complexity of this algorithm is bounded by $0(n^{2p+1})$. Note that for $p = 1$ the complexity of the algorithm presented in Drezner[14] is $0(n^2 \log n)$.

COMPUTATIONAL RESULTS

Demand points for all problems presented in this section were uniformly generated by a random number generator within a square. Equal weights were assumed. Run times are in seconds on the Amdahl 470/V8 computer at the University of Michigan, Ann Arbor. Programs were coded in FORTRAN IV-H. No special memory requirement was needed beyond the weights, locations of demand points and the matrix of $\{a_{ij}\}$ in equation (6).

In Table 1, run times for the heuristic procedures applied to large problems are presented.

In Table 2 run times of our method are compared with run times of the total enumeration approach, and the improvement in the value of the objective function of the optimal algorithm over that of the heuristic is given. Run times seem to be dependent on the number of iterations and the value of m in each iteration, two factors which are unknown *a priori*. In Drezner,[15] the special case of the two-centre problem is discussed. Run times for a different algorithm suggested there (for $p = 2$ only) are better than run times presented here by a factor of about 10.

REFERENCES

[1] R. L. LOVE and H. JUEL (1976) Properties and solution methods for large location allocation problems. M.R.C. Technical Summary Report No. 1634, University of Wisconsin–Madison.

[2] L. COOPER (1963) Location–allocation problems. *Opns Res.* **11,** 331–343.

[3] L. COOPER (1964) Heuristic methods for location–allocation problems. *SIAM Rev.* **6,** 37–52.

[4] L. COOPER (1972) The transportation–location problem. *Opns Res.* **20,** 94–108.

[5] S. EILON, C. D. T. WATSON-GANDY and N. CHRISTOPHIDES (1971) *Distribution Management: Mathematical Modelling and Practical Analysis.* Hafner, New York.

[6] Z. DREZNER and G. O. WESOLOWSKY (1980) Single facility l_p distance minimax location. *SIAM J. Algebraic Discrete Methods* **1,** 315–321.
[7] Z. DREZNER (1982) On minimax optimization problems. *Math. Programming* **22,** 227–230.
[8] M. I. SHAMOS and D. HOEY (1975) Closest point problems. *16th Annual I.E.E.E. Symposium on Foundations of Computer Science*, pp. 151–162.
[9] E. BALAS and M. W. PADBERG (1972) On the set covering problem. *Opns Res.* **20,** 1152–1161.
[10] R. S. GARFINKEL and G. L. NEMHAUSER (1972) *Integer Programming*. Wiley, New York.
[11] J. ELZINGA and D. W. HEARN (1972) Geometric solutions for some minimax location problems. *Transportation Sci.* **6,** 379–394.
[12] D. W. HEARN and J. VIJAY (1982) Efficient algorithms for the (weighted) minimum circle problem. *Opns Res.* **30,** 777–795.
[13] Z. DREZNER and B. GAVISH (1981) ε-Approximations for multi-dimensional weighted location problems. Working paper No. 8117, Graduate School of Management, University of Rochester, Rochester, New York.
[14] Z. DREZNER (1981) On a modified one-center model. *Mgmt Sci.* **27,** 848–851.
[15] Z. DREZNER (1981) On the two-center and two-median problems. Submitted for publication.

LINKED CITATIONS

References

[3] **Heuristic Methods for Location-Allocation Problems**
Leon Cooper
SIAM Review, Vol. 6, No. 1. (Jan., 1964), pp. 37-53.
Stable URL:
http://links.jstor.org/sici?sici=0036-1445%28196401%296%3A1%3C37%3AHMFLP%3E2.0.CO%3B2-Z