Management Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Exceptional Paper—Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms

Gerard Cornuejols, Marshall L. Fisher, George L. Nemhauser,

To cite this article:
Gerard Cornuejols, Marshall L. Fisher, George L. Nemhauser, (1977) Exceptional Paper—Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms. Management Science 23(8):789-810. https://doi.org/10.1287/mnsc.23.8.789

MANAGEMENT SCIENCE
Vol. 23, No. 8, April, 1977
Printed in U.S.A.

Exceptional Paper

LOCATION OF BANK ACCOUNTS TO OPTIMIZE FLOAT: AN ANALYTIC STUDY OF EXACT AND APPROXIMATE ALGORITHMS*†

GERARD CORNUEJOLS‡, MARSHALL L. FISHER§ AND GEORGE L. NEMHAUSER‡

The number of days required to clear a check drawn on a bank in city j depends on the city i in which the check is cashed. Thus, to maximize its available funds, a company that pays bills to numerous clients in various locations may find it advantageous to maintain accounts in several strategically located banks. We will discuss the problem of optimally locating bank accounts to maximize clearing times. The importance of this problem depends in part on its mathematical equivalence to the well-known uncapacitated plant location problem. We present a Lagrangian dual for obtaining an upper bound and heuristics for obtaining a lower bound on the value of an optimal solution. Our main results are analytical worst-case analyses of these bounds. In particular we show that the relative error of the dual bound and a "greedy" heuristic never exceeds $[(K-1)/K]^K < 1/e$ for a problem in which at most K locations are to be chosen. Two other heuristics are shown to have worst-case relative errors of at least $(K-1)/(2K-1) < \frac{1}{2}$. Examples are given showing that all these bounds can be achieved. We present extensive computational results for these approximations.

The number of days required to clear a check drawn on a bank in city j depends on the city i in which the check is cashed. Thus, to maximize its available funds, a company that pays bills to numerous clients in various locations may find it advantageous to maintain accounts in several strategically located banks. It would then pay bills to clients in city i from a bank in city $j(i)$ that had the largest clearing time. The economic significance to large corporations of locating accounts so that large clearing times can be achieved is discussed in an article in *Businessweek* [37].

To formalize the problem of selecting an optimal set of account locations, let $I = \{1, \ldots, m\}$ be the set of client locations, $J = \{1, \ldots, n\}$ the set of potential account locations, d_j the fixed cost of maintaining an account in city j, f_i the dollar volume of checks paid in city i, ϕ_{ij} the number of days (translated into monetary value) to clear a check issued in city j and cashed in city i, and K the maximum number of accounts that can be maintained. All of this information is assumed to be known and $c_{ij} = f_i\phi_{ij}$ represents the value of paying clients in city i from an account in city j.

Let

$$y_j = 1 \quad \text{if an account is maintained in city } j,$$
$$= 0 \quad \text{otherwise.}$$

It is optimal to pay all customers in city i from one account. Let $x_{ij} = 1$ if customers in city i are paid from account j, and $x_{ij} = 0$ otherwise. If $y_j = 0$ we must have $x_{ij} = 0$

* This paper has been refereed under guidelines for exceptional contributions.
† Accepted by Arthur M. Geoffrion; received February 7, 1976. This paper has been with the authors 2 months, for 1 revision.
‡ CORE, University of Louvain, Belgium. The research of these authors was supported by National Science Foundation Grant ENG75-00568 to Cornell University.
§ University of Pennsylvania. The research of this author was supported by National Science Foundation Grant SOC-7402516.

for all $i \in I$. Thus the account location problem, which we call (P), can be stated as the integer linear program (IP)

$$z = \max \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} - \sum_{j \in J} d_j y_j \tag{1}$$

$$\sum_{j \in J} x_{ij} = 1, \qquad i \in I, \tag{2}$$

$$1 \leqslant \sum_{j \in J} y_j \leqslant K, \tag{3}$$

$$0 \leqslant x_{ij} \leqslant y_j \leqslant 1, \qquad i \in I, j \in J, \tag{4}$$

$$x_{ij} \quad \text{and} \quad y_j \quad \text{integral}, \qquad i \in I, j \in J. \tag{5}$$

It is possible to obtain a more compact integer linear programming formulation (IP') of (P) by replacing the mn constraints $x_{ij} \leqslant y_j$, $i \in I, j \in J$, with

$$\sum_{i \in I} x_{ij} \leqslant m y_j, \qquad j \in J. \tag{6}$$

As will be seen later however, there are disadvantages in doing this.

The essential variables in (P) are the y_j's since given binary-valued y_j's, say $J^0 = \{j \mid y_j = 1\}$, it is simple to determine an optimal set of x_{ij}'s. Let $J^0(i) = \{j \in J^0 \mid c_{ij} = \max_{k \in J^0} c_{ik}\}$. Then, with respect to J^0, an optimal set of x_{ij}'s is given by $x_{ij} = 1$ for some $j \in J^0(i)$ and $x_{ij} = 0$ otherwise.

There is a vast literature on problems that are mathematically equivalent to problem (P). Besides desiring to delay payments for as long as possible, corporations also want to collect funds due them as quickly as possible. This can be done by situating check collection centers or "lock-boxes" at optimal locations. Problem (P), with a suitable redefinition of terminology, is also a model for the lock-box problem (Kraus, Janssen and McAdams [31], Maier and Vander Weide [36]).

When $I = J$ are the nodes of a graph and (1) and (3) are modified to $\min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$ and $\sum_{j \in J} y_j = K$, the model is known as the K-median problem. When (1) is modified to $\min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{j \in J} d_j y_j$, the model is known as the simple or uncapacitated plant or warehouse location problem. Francis and Goldstein [15] give a bibliography on location theory. Geoffrion [19] presents a user-oriented discussion of practical applications and methodology for location problems.

The first methodologies for these problems were heuristics. One simple heuristic forms a solution in a single pass by selecting locations sequentially in an order that maximizes the increase in the objective at each step. We will call this the *greedy heuristic* because of its appetite for immediate improvement. This heuristic appears in Kuehn and Hamburger [33], Spielberg [49], and Jarvinen et al. [23].

Another heuristic attempts to improve an initial feasible solution by interchanging a location currently in the solution with one that is not in the solution. This process continues until a solution is found that cannot be improved by such interchanges. This method, which we call the *interchange heuristic*, was described in [33], elaborated on by Manne [38], proposed for a more general class of combinatorial optimization problems by Reiter and Sherman [42], applied to the K-median problem by Teitz and Bart [51], and extended and evaluated empirically with extensive computations by Diehr [6]. Baker [1] has given a *dynamic programming heuristic* in which a DP formulation of (P) with a state-space of cardinality 2^n is approximated by a DP with a state-space of cardinality Kn. Other heuristics for this class of problems have been proposed and evaluated empirically by Feldman, Lehrer and Ray [11], Maranzana [39], Drysdale and Sandiford [7], Jarvinen et al. [23], Levy [34], and Cooper [3], [4].

Beginning in 1966 with the work of Effroymson and Ray [8], a number of papers on branch-and-bound methods have appeared. Various branching options for these algorithms based on selecting or eliminating a location (fixing a y_j at 0 or 1) are discussed by Khumawala [29]. In choosing a relaxation to obtain bounds there is the usual tradeoff between the tightness of the bounds and the amount of work required to obtain them. A standard relaxation for this class of problems is obtained by relaxing the integrality constraint on x_{ij} and y_j. This gives us two different linear programs (LP) and (LP′) corresponding to (IP) and (IP′) respectively. It is easy to see that every feasible solution to (LP) is also a feasible solution to (LP′), but that the converse is not true. For this reason we call (LP′) the weak linear programming relaxation. Problem (LP′) can be solved analytically and has been used by Effroymson and Ray [8], Spielberg [49], [50] and Khumawala [29]. Methods for structured linear programs have been developed to solve the strong relaxation (LP). Garfinkel et al. [17] apply Dantzig-Wolfe decomposition, Schrage [47] has developed a generalization of the simplex method with upper bound constraints to treat the variable upper bounds that appear in constraint (4), and Marsten [40] also has tailored the simplex method for solving (LP). Other branch-and-bound algorithms for problem (P) have been developed by El-Shaieb [9] and Jarvinen et al. [23].

Useful relaxations for a variety of combinatorial problems have been obtained by identifying a set of complicating constraints of the problem, weighting these constraints by multipliers and placing them in the objective function. This dual method, which has been called Lagrangian relaxation, has its roots in the seminal papers by Lorie and Savage [35] and Everett [10]. It was first shown to be a very effective computational tool for solving large combinatorial problems by Held and Karp [20], [21] in their work on the traveling salesman problem. Other successful applications are documented in Fisher [12], [13], Fisher and Shapiro [14], Shapiro [48], and Ross and Soland [44]. Geoffrion [18] gives a lucid exposition of the theory and practical uses of Lagrangian relaxation. Geoffrion [18] has proposed a Lagrangian relaxation for (P) in which one dualizes (IP) with respect to the constraints (2). This relaxation is also given in a different manner by Diehr [6]. This partial dual is intimately related to the strong linear program (LP). For example, both problems have the same optimum objective value.

Although most recent work on the class of problems represented generically by problem (P) has been on exact algorithms, there is still a need to study heuristics. Heuristics provide feasible solutions and lower bounds for exact algorithms. Most importantly, however, heuristics appear to be the only reasonable option for solving very large problems. The reason for this pessimistic remark is that problem (P) is easily shown to be NP-complete, which means that any problem in the class of NP-complete problems is polynomially reducible to (P), see Cook [2] and Karp [27], [28]. The result that (P) is NP-complete is easily established by reducing the NP-complete node covering problem (see Karp [28]) to (P). This fact means that an algorithm for problem (P) with a running time bound that is a polynomial in the parameters m, n and K exists if and only if virtually all combinatorial optimization problems, including the traveling salesman problem and 0-1 integer programming problems, can be solved in polynomial time. Thus, it is very unlikely that an efficient algorithm exists for solving (P).

Traditionally, the running time of an exact algorithm and the quality of solutions produced by an approximate algorithm for (P) have been evaluated by computational testing on a selected sample of data sets. This approach has some difficulties. A potential user of the algorithm might be concerned that the sample of test problems omits those data sets of most interest to him, and even if the sample were selected to give an accurate estimate of average performance, there is always the danger of poor

performance on a particular set of data. Also, computational testing frequently fails to provide insight into why an algorithm performs poorly in some cases.

This paper contains an analytic study of exact and approximate algorithms for problem (P). The results of such a study can be used in conjunction with computational testing to alleviate the difficulties cited above. The type of result we will derive is most easily illustrated when all $d_j = 0$ and all $c_{ij} \geqslant 0$ in problem (P). In this case we show, for example, that the value of the solution obtained with the greedy heuristic is at least $(e-1)/e$ times the optimal value of (LP), where $e = 2.718\ldots$ is the base of the natural logarithm. Since the optimal value of (LP) equals the optimal value of the Lagrangian dual this result implies upper bounds on the relative errors of both the greedy heuristic and the (LP) and Lagrangian relaxations. We will show that these bounds are tight. Although the study of heuristic error bounds is developing rapidly (see Johnson [24] for a recent survey), there are very few cases for which a (nontrivial) tight bound is known for the relative difference between the optimal value of a family of integer programs and its corresponding linear programming relaxations.

We have also conducted computational tests on the various heuristics studied analytically and on the Lagrangian relaxation. Although our primary objective has not been to develop a single algorithm for the location problem, the best combinations of algorithmic components we tested have performed better than existing algorithms on large test problems obtained both from the literature and real applications.

The paper is organized into five sections. In §1 we give a criterion for measuring heuristics and relaxations and describe some of the other work in this area. §2 describes the Lagrangian relaxation and its relation to the linear programming relaxations. We also formally define the greedy heuristic in this section. §3 contains our most important result, a tight upper bound on the worst performance of the greedy heuristic and the Lagrangian and (LP) relaxations. In §4 we analyze theoretically the interchange heuristic and the dynamic programming heuristic of Baker [1]. Although these heuristics are computationally more expensive than the greedy heuristic, we will show that their worst possible performances are inferior to those of the greedy heuristic. Finally, §5 contains the computational results that complement our theoretical analyses.

1. A Criterion for Measuring the Quality of Bounds

Let $\mathcal{P}$ be the family of problems generated from problem (P) by considering all positive integer values for m, n and K, all real $m \times n$ matrices $C = \{c_{ij}\}$ and all real n-vectors $d = (d_1, \ldots, d_n)$. As before, z denotes the optimal objective value of a particular $P \in \mathcal{P}$. Now let $\bar{z}$ and $\underline{z}$ be upper and lower bounds respectively on z. These bounds may be obtained, for example, from the strong linear programming relaxation and greedy heuristic, respectively.

When evaluating the quality of a bound—for definiteness say a lower bound—it is not in general meaningful to consider the absolute deviation $z - \underline{z}$, since this deviation is sensitive to scale changes in the data. Thus if there is a (P) that yields a positive absolute deviation, we can construct problems in $\mathcal{P}$ with arbitrarily large deviations.

Relative deviations are more meaningful. However, defining an appropriate measure of relative deviation is subtle. For example, a popular measure of relative deviation for a heuristic in a maximization problem is $F = (z - \underline{z})/z = 1 - \underline{z}/z$. This measure is appropriate when $\underline{z} \geqslant 0$ and $z > 0$ in which case $0 \leqslant F \leqslant 1$. For a particular heuristic one seeks to show that $F \leqslant \epsilon < 1$ for all problems within some class. This is equivalent to showing that the ratio $z/\underline{z}$ is bounded by the positive constant $1/(1-\epsilon)$. The measure F has been used by Sahni [45] in analyzing a heuristic for the knapsack problem with positive data. Johnson [24], in a recent survey

of heuristics for combinatorial optimization problems with positive data, has focused on bounds on $z/\underline{z}$ for maximization problems and $\underline{z}/z$ for minimization problems.

The measure F is inadequate for our problem. We cannot require $z > 0$ since a minimization problem such as the simple plant location or K-median problem, when translated into a maximization problem, would generally have $z < 0$. More importantly, the measure F for our problem fails to have the following property that we believe is essential. A modification of the data that adds a constant to the objective value of every feasible solution but leaves the execution of the heuristic unchanged should also leave the error measure unchanged. For example, if a constant δ is added to every element of a row of C in problem (P), then the objective value of each feasible solution is increased by δ, but the execution of the greedy heuristic (among others) is unchanged. The measure F is now equal to $(z - \underline{z})/(z + \delta)$ and, provided $z \neq \underline{z}$, it can be made as large (or small) as we like by appropriate choice of δ. Indeed, it is easy to show by reasoning similar to that used in [46] that the problem of finding a solution for which $F \leqslant \epsilon < 1$ for fixed ϵ is itself NP-complete.

With these considerations in mind, to evaluate lower bounds obtained from a heuristic we use the measure $G = (z - \underline{z})/(z - z_R)$, where z_R is a suitably chosen reference value for (P). Ideally, the reference z_R should equal the minimum objective value of (P) but, in any event, z_R should be a lower bound on this minimum value that is sensitive to significant data changes such as the addition of a constant to every element of a row of C. We may think of $z - z_R$ as the worst absolute deviation that could be achieved by a heuristic. Then G measures the deviation for a particular heuristic relative to the worst possible deviation.

In problem (P) we define

$$z_R = \sum_{i \in I} \min_{j \in J} c_{ij} - K\Big(\max_{j \in J} d_j\Big).$$

If $z_R = 0$, $G = F$ and if $z_R > 0$, $F \leqslant G$ so that a bound on G implies a bound on F.

Our measure for evaluating upper bounds in maximization problems is similar to G. Using the same value for z_R, we define an error measure of an upper bound to be $H = (\bar{z} - z)/(\bar{z} - z_R)$.

We will assume that $\mathcal{P}$ has been restricted to exclude all problems for which $z - z_R = 0$ or $\bar{z} - z_R = 0$. The relations $z_R \leqslant \underline{z} \leqslant z \leqslant \bar{z}$ would make error bound analysis rather pointless in these cases. We note that $0 \leqslant G \leqslant 1$, $0 \leqslant H \leqslant 1$, $G = 0$ if and only if $\underline{z} = z$, and $H = 0$ if and only if $\bar{z} = z$.

There are two basic approaches for extending the measures from a particular (P) to the family $\mathcal{P}$. One is statistical and the other is a worst-case analysis. Practically, we might like to know statistics such as the mean and variance of the relative deviation over $\mathcal{P}$. However, to obtain such results we would have to assume a probability distribution over the problems in $\mathcal{P}$. Furthermore, analyses of this kind appear to be very difficult mathematically. Worst-case analysis seems to be more fruitful. Johnson et al. [25] give a very successful worst-case analysis of the bin-packing problem. Some other references, besides the survey of Johnson [24] cited earlier, are Sahni [45], Garey and Graham [16] and Rosenkrantz et al. [43].

We call a heuristic "good" if $\sup_{P \in \mathcal{P}} G$ is smaller than one. Similarly we call a relaxation "good" if $\sup_{P \in \mathcal{P}} H$ is smaller than one. The distinction between "good" and "bad" bounds is by no means trivial since many heuristics and relaxations are not good. For example, the greedy heuristic for the 0-1 knapsack problem in which the items are put into the knapsack according to decreasing value/size ratio is not good (Sahni [45]). There are no known "good" heuristics that run in polynomial time for the set covering and maximum clique problems (Johnson [24]) and the heuristic of Levy [34] for the problem considered here is not good.

Regarding upper bounds, the linear programming relaxation of an integer program is not, in general, good. For example, consider the integer programming problem of finding a maximum independent set of nodes in a graph (Nemhauser and Trotter [41]). For a complete graph of n nodes, the cardinality of a maximum independent set is equal to one. But, with respect to the standard formulation with the constraints $x_i + x_j \leqslant 1$ if nodes i and j are connected, the linear programming relaxation yields a value of $n/2$. We also will show in §2 that the weak linear program (LP′) is not good.

Finally, in comparing approximation algorithms that produce "good" bounds, we must consider the running time as well as the quality of the bound. One might want to consider only heuristics (or relaxations) that run in polynomial time (Johnson [24]). Again, not all methods meet this criterion. For example, there is no known polynomial bound on the running time of the widely-used interchange heuristic for (P) that we consider in §4.

2. A Lagrangian Relaxation and the Greedy Heuristic

Geoffrion [18] has proposed a Lagrangian relaxation for problem (P) in which the constraints (2) are weighted by multipliers and placed in the objective function. Let x and y be vectors of x_{ij}, $i \in I$, $j \in J$, and y_j, $j \in J$. Let $u = (u_1, \ldots, u_m)$ be multipliers for constraints (2) and define

$$L(x, y, u) = \sum_{i \in I} \sum_{j \in J} c_{ij}x_{ij} - \sum_{j \in J} d_j y_j - \sum_{i \in I} u_i \left(\sum_{j \in J} x_{ij} - 1 \right)$$

$$= \sum_{j \in J} \left[\sum_{i \in I} (c_{ij} - u_i)x_{ij} - d_j y_j \right] + \sum_{i \in I} u_i.$$

The Lagrangian problem is given by $z_D(u) = \max_{(x, y)} L(x, y, u)$ subject to (3), (4), (5), and the corresponding Lagrangian dual by $z_D = \min_u z_D(u)$. It is well known that $z_D \geqslant z$. Furthermore, it is easy to show by direct consideration of square submatrices that the coefficient matrix corresponding to constraints (3) and (4) is totally unimodular. It then follows from Theorem 2 of Geoffrion [18] that z_D is equal to the optimum value of the strong linear programming relaxation (LP).

It is easy to solve the Lagrangian problem analytically to determine $z_D(u)$ for fixed u. From the second expression for $L(x, y, u)$ and (4) we note that the optimal values for x_{ij}'s in the Lagrangian problem are given by

$$x_{ij} = y_j \quad \text{if } c_{ij} - u_i \geqslant 0,$$
$$= 0 \quad \text{otherwise.}$$

Hence defining

$$\rho_j(u) = \sum_{i \in I} \max(0, c_{ij} - u_i) - d_j$$

optimal y_j's must solve

$$\max \sum_{j \in J} \rho_j(u) y_j \quad \text{subject to } 1 \leqslant \sum_{j \in J} y_j \leqslant K, \qquad y_j = 0 \text{ or } 1, j \in J.$$

Define $J^+(u) = \{j \in J \mid \rho_j(u) > 0\}$ and set $J(u) = J^+(u)$ if $1 \leqslant |J^+(u)| \leqslant K$. Otherwise let $J(u)$ be an index set corresponding to the K largest $\rho_j(u)$ if $|J^+(u)| > K$ or the single largest $\rho_j(u)$ if $|J^+(u)| = 0$.

PROPOSITION 1. *An optimal solution to the Lagrangian problem is given by $y_j = 1$ if $j \in J(u)$, $y_j = 0$ otherwise; $x_{ij} = 1$ if $y_j = 1$ and $c_{ij} - u_i > 0$, and $x_{ij} = 0$ otherwise.*

As a consequence of Proposition 1 we have that

$$z_D(u) = \sum_{j \in J(u)} \rho_j(u) + \sum_{i \in I} u_i. \tag{7}$$

(7) will play a central role in the analysis of §3 and §4.

When $d \geqslant 0$ it is possible to relate the Lagrangian dual to the weak linear program (LP') by specifying a $\bar{u}$ ($\bar{u}$ does not in general minimize $z_D(u)$) such that $z_D(\bar{u}) \leqslant z_{\mathrm{LP}'}$, where $z_{\mathrm{LP}'}$ equals the optimal value of (LP'). An optimal solution to (LP') is given by $x_{ij(i)} = 1$, $i \in I$, where $j(i)$ is a location satisfying $c_{ij(i)} - d_{j(i)}/m = \max_{j \in J}(c_{ij} - d_j/m)$, $x_{ij} = 0$, otherwise; $y_j = \sum_{i \in I} x_{ij}/m$ for all $j \in J$; and $z_{\mathrm{LP}'} = \sum_{i \in I}(c_{ij(i)} - d_{j(i)}/m)$—see Khumawala [29].

PROPOSITION 2. *For each $i \in I$, define $\bar{u}_i = c_{ij(i)} - d_{j(i)}/m$. Then for $d \geqslant 0$, $z_D(\bar{u}) \leqslant z_{\mathrm{LP}'}$.*

PROOF. We have $\sum_{i \in I} \bar{u}_i = z_{\mathrm{LP}'}$. Then by (7), it suffices to show that $\rho_j(\bar{u}) \leqslant 0$ for all $j \in J$. By definition

$$\begin{aligned} \rho_j(\bar{u}) &= \sum_{i \in I} \left[\max(0,\, c_{ij} - c_{ij(i)} + d_{j(i)}/m) - d_j/m\right] \\ &= \sum_{i \in I} \max\left(-d_j/m,\, c_{ij} - d_j/m - (c_{ij(i)} - d_{j(i)}/m)\right) \leqslant 0, \end{aligned}$$

where the inequality holds because $d_j \geqslant 0$ and the definition of $j(i)$ implies $c_{ij} - d_j/m - (c_{ij(i)} - d_{j(i)}/m) \leqslant 0$.

PROPOSITION 3. *The weak linear programming relaxation* (LP') *is not "good."*

PROOF. Define $H_{\mathrm{LP}'} = (z_{\mathrm{LP}'} - z)/(z_{\mathrm{LP}'} - z_R)$ and consider the subfamily of $\mathcal{P}$ with $m = n$, $d = 0$, $C = I_n$ (an $n \times n$ identity matrix) and $K = 1$. Then for all (P) in the subfamily, we have $z_R = 0$, $z = 1$ and $z_{\mathrm{LP}'} = n$. (Set $x_{ii} = 1$ and $y_i = 1/n$, $i = 1, \ldots, n$.) Thus $H_{\mathrm{LP}'} = (n-1)/n$ and $\sup_{P \in \mathcal{P}} H_{\mathrm{LP}'} = 1$.

We conclude this section with a formal statement of the greedy heuristic. The greedy heuristic first chooses a location that solves (P) when $K = 1$ and then proceeds recursively. Suppose $k < K$ locations have been selected. If there exists an unselected location that improves the value of the objective function, choose one that yields the maximum improvement; otherwise stop. Since our analysis of this heuristic will depend heavily on the Lagrangian relaxation just developed, we will state the heuristic in the notation of this development. We also define quantities u_i^k and ρ_k that will be useful in our subsequent analysis.

The Greedy Heuristic

Step 1. Let $k = 1$, $J^* = \emptyset$ and $u_i^1 = \min_{j \in J} c_{ij}$, $i \in I$.

Step 2. Let $\rho_j(u^k) = \sum_{i \in I} \max(0,\, c_{ij} - u_i^k) - d_j$, $j \notin J^*$. Find $j_k \notin J^*$ such that $\rho_k = \rho_{j_k}(u^k) = \max_{j \notin J^*} \rho_j(u^k)$. If $\rho_k < 0$ and $|J^*| \geqslant 1$ set $k = k - 1$ and go to Step 4. Otherwise set $J^* = J^* \cup \{j_k\}$. If $|J^*| = K$ go to Step 4, otherwise go to Step 3.

Step 3. Set $k = k + 1$. For $i \in I$ set $u_i^k = \max_{j \in J^*} c_{ij} = u_i^{k-1} + \max(0,\, c_{ij_{k-1}} - u_i^{k-1})$. Go to Step 2.

Step 4. Stop; the greedy solution is given by $y_j = 1$, $j \in J^*$, and $y_j = 0$ otherwise. We have $|J^*| = k$ and the value of the greedy solution is $z_g = \sum_{i=1}^m u_i^1 + \sum_{j=1}^k \rho_j$.

The following example illustrates the greedy heuristic with $d = 0$, $K = 2$ and

$$C = \begin{bmatrix} 0 & 11 & 6 & 9 \\ 7 & 0 & 8 & 2 \\ 7 & 3 & 0 & 3 \\ 10 & 9 & 4 & 0 \end{bmatrix}.$$

We initialize with $J^* = \emptyset$ and $u^1 = (0, 0, 0, 0)$. Then $\rho_1(u^1) = 24$, $\rho_2(u^1) = 23$, $\rho_3(u^1) = 18$, $\rho_4(u^1) = 14$, $j_1 = 1$ and $J^* = \{1\}$. We set $u^2 = (0, 7, 7, 10)$ and obtain $\rho_2(u^2) = 11$, $\rho_3(u^2) = 7$ and $\rho_4(u^2) = 9$. Thus $j_2 = 2$, $J^* = \{1, 2\}$, and $z_g = 35$. We also note that $u^3 = (11, 7, 7, 10)$ and $z_D(u^3) = 1 + 35 = 36$ from (7) so that $35 \leqslant z \leqslant 36$.

3. Analysis of the Greedy Heuristic and Lagrangian Dual

In this section we show that

$$(z_D - z_g)/(z_D - z_R) < 1/e \quad \text{for all } P \in \mathcal{P}. \tag{8}$$

Since $z_R \leqslant z_g \leqslant z \leqslant z_D$, (8) implies that

$$G_g = (z - z_g)/(z - z_R) < 1/e \quad \text{and} \quad H_D = (z_D - z)/(z_D - z_R) < 1/e.$$

We will present examples to show that these are the best possible bounds; that is, $\sup_{P \in \mathcal{P}} G_g = \sup_{P \in \mathcal{P}} H_D = 1/e$. Furthermore, since $z_D = z_{\mathrm{LP}}$, the optimal value of the strong linear programming relaxation (LP), we also obtain the result that $(z_{\mathrm{LP}} - z)/(z_{\mathrm{LP}} - z_R) < 1/e$.

Since our derivation will at times be quite intricate, we will motivate the results by first considering the case where $C \geqslant 0$, $\min_{j \in J} c_{ij} = 0$, $i \in I$, and $d = 0$ so that $z_R = 0$. In this case we have $u^1 = 0$, K locations are selected,

$$z_g = \sum_{k=1}^{K} \rho_k, \tag{9}$$

and our objective is to determine an upper bound on $1 - z_g/z_D$. We will first establish the comparatively weak bound $1 - z_g/z_D < \frac{1}{2}$ and then show how it can be improved.

Note that $u^1 = 0$ implies $\sum_{i \in I} u_i^K = \sum_{k=1}^{K-1} \rho_k$. Furthermore, $\rho_K \geqslant \rho_j(u^K)$, $j \in J$. Substituting these facts into (7) gives

$$z_D \leqslant z_D(u^K) \leqslant \sum_{k=1}^{K-1} \rho_k + K\rho_K. \tag{10}$$

From (9) and (10) it follows that

$$1 - z_g/z_D \leqslant 1 - \sum_{k=1}^{K} \rho_k \Big/ \left[\sum_{k=1}^{K-1} \rho_k + K\rho_K \right] = (K-1)\rho_K \Big/ \left[\sum_{k=1}^{K-1} \rho_k + K\rho_K \right]. \tag{11}$$

Now from $u_i^k \geqslant u_i^{k-1}$ we obtain $\rho_k \leqslant \rho_{k-1}$ so that $\sum_{k=1}^{K-1} \rho_k \geqslant (K-1)\rho_K$. This result and (11) imply $1 - z_g/z_D \leqslant (K-1)/(2K-1) < \frac{1}{2}$ for all K.

The bound of $(K-1)/(2K-1)$ is not tight because in deriving it we ignored upper bounds on z_D that can be obtained from the multipliers $u^1, \ldots, u^{K-1}$ of the greedy heuristic. For example, for $K = 2$ two bounds on z_D are available—$2\rho_1$ from u^1 and $\rho_1 + 2\rho_2$ from u^2. If $\rho_1 \leqslant 2\rho_2$ the first bound is smallest and we have

$$1 - z_g/z_D \leqslant (2\rho_1 - \rho_1 - \rho_2)/2\rho_1 \leqslant \tfrac{1}{2}\rho_1/2\rho_1 = \tfrac{1}{4}.$$

If $\rho_1 \geqslant 2\rho_2$ then

$$1 - z_g/z_D \leqslant (\rho_1 + 2\rho_2 - \rho_1 - \rho_2)/(\rho_1 + 2\rho_2) \leqslant \rho_2/4\rho_2 = \tfrac{1}{4}.$$

The case $\rho_1 = 2\rho_2$ is critical since it can be shown that $1 - z_g/z_D = \frac{1}{4}$ if and only if $\rho_1 = 2\rho_2$.

For general k, analogous to (10), we have

$$z_D \leqslant z_D(u^k) \leqslant \sum_{l=1}^{k-1} \rho_l + K\rho_k, \qquad k = 1, \ldots, K. \tag{12}$$

It can be shown that the critical case occurs when all K of these bounds from (12) are equal and that equality is obtained when $\rho_k = [(K-1)/K]\rho_{k-1}$, $k = 2, \ldots, K$. In this case

$$z_g = \sum_{k=1}^{K} \rho_k = \rho_1 \sum_{k=0}^{K-1} [(K-1)/K]^k = \rho_1 \left[\frac{1 - [(K-1)/K]^K}{1 - (K-1)/K} \right]$$

$$= K\rho_1 \left[1 - [(K-1)/K]^K \right].$$

Hence

$$1 - z_g/z_D \leqslant 1 - \frac{K\rho_1 \left[1 - [(K-1)/K]^K \right]}{K\rho_1} = [(K-1)/K]^K.$$

The sequence $[(K-1)/K]^K$, $K = 1, 2, \ldots,$ converges monotonically from below to $1/e$.

We will now give a formal proof of this result for all problems in the family $\mathcal{P}$. Let $\mathcal{P}_K$ be the subfamily of $\mathcal{P}$ in which at most K locations may be selected and let $\alpha = (K-1)/K$.

LEMMA 1. *For all* $P \in \mathcal{P}_K$, $(z_D - z_g)/(z_D - z_R) \leqslant \alpha^K = [(K-1)/K]^K$.

PROOF. We first establish the lemma when $d \leqslant 0$ and $\max_{j \in J} d_j = 0$ so that $z_R = \sum_{i \in I} u_i^1$. In this case K locations are selected by the greedy heuristic.

For $s = 1, \ldots, K$, $\sum_{i \in I} u_i^s \leqslant z_R + \sum_{j=1}^{s-1} \rho_j$, where the sum is zero if $s = 1$, and $\rho_s \geqslant \rho_j(u^s) \geqslant 0$, $j \in J$. Substituting these facts into (7) gives

$$z_D \leqslant z_R + \sum_{j=1}^{s-1} \rho_j + K\rho_s, \qquad s = 1, \ldots, K. \tag{13}$$

Multiply inequality s of (13) by $(1-\alpha)\alpha^{K-s}$ and sum for $s = 1, \ldots, K$. Noting that $\sum_{s=1}^{K} \alpha^{K-s} \sum_{j=1}^{s-1} \rho_j = \sum_{s=1}^{K-1} \sum_{j=0}^{K-s-1} \alpha^j \rho_s$ we obtain

$$(1-\alpha) \sum_{s=1}^{K} \alpha^{K-s} \left(\sum_{j=1}^{s-1} \rho_j + K\rho_s \right) = (1-\alpha) \sum_{s=1}^{K} \left(\sum_{j=0}^{K-s-1} \alpha^j + K\alpha^{K-s} \right) \rho_s.$$

Now $\sum_{j=0}^{K-s-1} \alpha^j + K\alpha^{K-s} = (1-\alpha)^{-1}$ and

$$(z_D - z_R)(1-\alpha) \sum_{s=1}^{K} \alpha^{K-s} = (z_D - z_R)(1 - \alpha^K).$$

Thus this weighted sum of the K inequalities of (13) yields $(z_D - z_R)(1 - \alpha^K) \leqslant \sum_{j=1}^{K} \rho_j$. Substituting $z_g - z_R = \sum_{j=1}^{K} \rho_j$ into this last inequality yields $(z_D - z_g)/(z_D - z_R) \leqslant \alpha^K$.

Now consider problem (P) for general d. Suppose that $\bar{u}$, $\bar{x}$ and $\bar{y}$ satisfy $z_D = z_D(\bar{u}) = L(\bar{x}, \bar{y}, \bar{u})$. Let $K' = \sum_{j \in J} \bar{y}_j$ and define a new problem (P') by replacing the fixed costs in (P) by $d_j' = d_j - \max_{j \in J} d_j$, $j \in J$, and K by K'. Since $d' \leqslant 0$ and $\max_{j \in J} d_j' = 0$, $(z_D' - z_g')/(z_D' - z_R') \leqslant \alpha^K$. Let $D = \max_{j \in J} d_j$. We will establish the lemma by showing that $z_R' \geqslant z_R + K'D$, $z_D' = z_D + K'D$ and $z_g' \leqslant z_g + K'D$. If $D \geqslant 0$ then $z_R' = z_R + KD \geqslant z_R + K'D$ follows from $K \geqslant K'$, while if $D < 0$ then $K = K'$ and $z_R' = z_R + KD$.

To see that $z_D' = z_D + K'D$ note that $z_D(u) = \min_u z_D''(u)$, where

$$z_D''(u) = \max_y \sum_{j \in J} \rho_j(u) y_j + \sum_{i \in I} u_i \quad \text{subject to} \quad \sum_{j \in J} y_j = K', \qquad y_j = 0 \text{ or } 1, \quad j \in J.$$

Furthermore, $\rho_j'(u) = \rho_j(u) - D \geqslant 0$ for all $j \in J$. Thus an optimal solution to the

Lagrangian problem for P′ can always have $\sum_{j\in J} y_j = K'$ and therefore $z''_D(u) = z'_D(u) - K'D$.

Finally, to show $z'_g \leqslant z_g + K'D$, define quantities $\rho_1, \ldots, \rho_{K'}$ by applying the greedy algorithm to (P) for K' iterations regardless of whether $\rho_k < 0$ in Step 2. Since $\rho'_j(u^k) = \rho_j(u^k) + D$, the order in which locations are selected is the same when greedy is applied to (P′) and

$$\begin{aligned} z'_g &= \sum_{i=1}^{m} u_i^1 + \rho'_1 + \cdots + \rho'_{K'} \\ &= \sum_{i=1}^{m} u_i^1 + \rho_1 + \cdots + \rho_{K'} + K'D \leqslant z_g + K'D, \end{aligned}$$

where the last inequality follows from the greedy choice rule and the fact that $K' \leqslant K$. Q.E.D.

As immediate consequences of Lemma 1 and the relations $z_R \leqslant z_g \leqslant z \leqslant z_D$ we have

THEOREM 1. *For all* $P \in \mathcal{P}_K$, $G_g \leqslant [(K-1)/K]^K$.

THEOREM 2. *For all* $P \in \mathcal{P}_K$, $H_D \leqslant [(K-1)/K]^K$.

We now give two families of problems in $\mathcal{P}_K$, $K = 2, 3, \ldots$, which show that $[(K-1)/K]^K$ is a tight bound for G_g and H_D respectively. ($K = 1$ is trivial.) Either family also implies the tightness of the bound in Lemma 1.

THEOREM 3. *Let* $P \in \mathcal{P}_K$, $K = 2, 3, \ldots$, *be defined by* $m = K(K-1)$, $n = 2K-1$, $d = 0$ *and* C^K, *where for* $j = 1, \ldots, K-1$

$$\begin{aligned} c_{ij}^K &= (K-1)K^{K-2}\alpha^{j-1} \qquad [\alpha = (K-1)/K], \qquad i = (j-1)K+1, \ldots, jK, \\ &= 0 \quad \textit{otherwise,} \end{aligned}$$

and for $j = K, \ldots, 2K-1$

$$\begin{aligned} c_{ij}^K &= K^{K-1}, \qquad i = 1 + j + (l-2)K, \quad l = 1, \ldots, K-1, \\ &= 0 \quad \textit{otherwise.} \end{aligned}$$

Then $G_g = [(K-1)/K]^K$, $K = 2, 3, \ldots$. (*Figure* 1 *shows* C^2, C^3 *and* C^4.)

$$C^2 = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 0 & 2 \end{bmatrix},$$

$$C^3 = \begin{bmatrix} 6 & 0 & 9 & 0 & 0 \\ 6 & 0 & 0 & 9 & 0 \\ 6 & 0 & 0 & 0 & 9 \\ 0 & 4 & 9 & 0 & 0 \\ 0 & 4 & 0 & 9 & 0 \\ 0 & 4 & 0 & 0 & 9 \end{bmatrix},$$

$$C^4 = \begin{bmatrix} 48 & 0 & 0 & 64 & 0 & 0 & 0 \\ 48 & 0 & 0 & 0 & 64 & 0 & 0 \\ 48 & 0 & 0 & 0 & 0 & 64 & 0 \\ 48 & 0 & 0 & 0 & 0 & 0 & 64 \\ 0 & 36 & 0 & 64 & 0 & 0 & 0 \\ 0 & 36 & 0 & 0 & 64 & 0 & 0 \\ 0 & 36 & 0 & 0 & 0 & 64 & 0 \\ 0 & 36 & 0 & 0 & 0 & 0 & 64 \\ 0 & 0 & 27 & 64 & 0 & 0 & 0 \\ 0 & 0 & 27 & 0 & 64 & 0 & 0 \\ 0 & 0 & 27 & 0 & 0 & 64 & 0 \\ 0 & 0 & 27 & 0 & 0 & 0 & 64 \end{bmatrix}.$$

FIGURE 1. Examples for Theorem 3.

PROOF. Clearly $z_R = 0$. The last K columns are an optimal set of locations and $z = K(K-1)K^{K-1} = K^K(K-1)$.

We now show that the first K locations can be selected in natural order in a solution produced by the greedy heuristic. We reason inductively. On the first iteration $\rho_1(u^1) = (K-1)K^{K-1}\alpha^0$, $\rho_j(u^1) = (K-1)K^{K-1}\alpha^{j-1}$, $j = 2, \ldots, K-1$, and $\rho_j(u^1) = (K-1)K^{K-1}$, $j = K, \ldots, 2K-1$. Hence $\rho_1(u^1) \geqslant \rho_j(u^1)$, $j \in J$, and, allowing arbitrary tie breaking, location 1 can be selected first.

Now assume that locations $1, \ldots, s-1$ have been selected on the first $s-1$ iterations. If $s = K$ then by symmetry any of the last K columns may be selected. If $s < K$ then $\rho_s(u^s) = (K-1)K^{K-1}\alpha^{s-1}$, $\rho_j(u^s) = (K-1)K^{K-1}\alpha^{j-1}$, $j = s+1, \ldots, K-1$, and for $j = K, \ldots, 2K-1$

$$\rho_j(u^s) = (K-1)K^{K-1} - (K-1)K^{K-2}\sum_{i=0}^{s-2}\alpha^i = (K-1)K^{K-1}\alpha^{s-1}.$$

Hence $\rho_s(u^s) \geqslant \rho_j(u^s)$ for all $j \geqslant s$ and location s can be selected on iteration s.

The corresponding objective value is

$$z_g = (K-1)K^{K-1}\sum_{j=0}^{K-1}\alpha^j = (K-1)\left(K^K - (K-1)^K\right).$$

Thus

$$G_g = \frac{(K-1)K^K - (K-1)\left[K^K - (K-1)^K\right]}{(K-1)K^K - 0} = \left[(K-1)/K\right]^K. \quad \text{Q.E.D.}$$

We note that if one prefers problems in which the greedy solution is unique then simply add a sufficiently small positive constant (ϵ) to elements c_{i1}, $i = 1, \ldots, K$, which yields $G_g = [(K-1)/K]^K - \epsilon/K^K$.

THEOREM 4. *Let* $P_t \in \mathcal{P}_K$, $t = 2, 3, \ldots$, $K = 2, 3, \ldots$, *be defined by* $d = 0$, $n = Kt$, $m = \binom{n}{t}$ *and the* 0-1 *matrix* C^{Kt}, *where the rows of* C^{Kt} *consist of all* 0-1 *n-vectors with precisely t positive elements. Then* $z_R = 0$, $z_D = m = \binom{n}{t}$, $z = \sum_{s=1}^{K}\binom{n-s}{t-1}$ *and, for each K, as t approaches infinity* H_D *approaches* $[(K-1)/K]^K$. (*Figure* 2 *shows* C^{22} *and* C^{23}.)

$$C^{22} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \qquad C^{23} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

FIGURE 2. Examples for Theorem 4.

PROOF. It is obvious that $z_R = 0$ for each t and K.

A feasible solution to (LP) with value m is given by $y_j = 1/t$, $j \in J$, and

$$x_{ij} = y_j \quad \text{if } c_{ij}^{Kt} = 1,$$
$$= 0 \quad \text{otherwise.}$$

To establish the optimality of this solution note that for any feasible solution to (LP) we have $\sum_{j\in J} c_{ij}^{Kt} x_{ij} \leqslant 1$ for all i since $\sum_{j\in J} x_{ij} = 1$. Thus $z_{\mathrm{LP}} = \max \sum_{i\in I}\sum_{j\in J} c_{ij}^{Kt} x_{ij} \leqslant m$ and $z_D = z_{\mathrm{LP}} = m$.

From the symmetry of C^{Kt} any selection of K locations gives the same objective value in problem (P_t). Assume that the rows of C^{Kt} are arranged in lexicographically decreasing order and that the first K locations are selected. The first column of C^{Kt} contains $\binom{n-1}{t-1}$ ones. In the second column a one appears in $\binom{n-2}{t-1}$ rows that do not contain a one in the first column; similarly in column s a one appears in $\binom{n-s}{t-1}$ rows that do not contain a one in the first $s-1$ columns. Hence $z = \sum_{s=1}^{K} \binom{n-s}{t-1}$.

Then using a known combinatorial identity we obtain $z = \binom{n}{t} - \binom{n-K}{t}$ so that

$$H_D = \frac{(z_D - z)}{(z_D - z_R)} = \frac{\binom{n-K}{t}}{\binom{n}{t}}$$
$$= \frac{(n-K)!\,(n-t)!}{n!\,(n-K-t)!} = \prod_{i=0}^{K-1} \left[\frac{n-t-i}{n-i} \right].$$

Substituting $n = Kt$ into this last expression yields

$$H_D = \prod_{j=0}^{K-1} \left[\frac{(K-1)t - i}{Kt - i} \right]$$

Noting that

$$\lim_{t\to\infty} \left[\frac{(K-1)t - i}{Kt - i} \right] = \left[\frac{K-1}{K} \right]$$

we obtain $\lim_{t\to\infty} H_D = [(K-1)/K]^K$. Q.E.D.

We close this section by considering an analogue of the greedy heuristic that begins with the n locations in J and sequentially drops locations according to the rule, "Drop a location whose elimination will cause the smallest decrease in the objective" (see [11]). This heuristic is not good. For example, consider the problem with $K = 2$, $d = 0$, $n = m + 1 \geqslant 4$ and

$$C = \begin{bmatrix} 1 & 2 & 0 & \cdots & 0 \\ 1 & 0 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 2 \end{bmatrix}.$$

We have $z_R = 0$ and $z = 2 + (m-1)$. With all n columns included, the value of the objective is $2m$. The penalty of deleting the first location is zero, and the penalty of deleting each of the other locations is one. Thus location 1 is deleted first. After this, any of the other locations can be deleted in any order and we obtain a solution of value 4. The relative error associated with this solution is $[2 + (m-1) - 4]/(2 + m - 1) = (m-3)/(m+1)$. This relative error approaches one as m approaches infinity.

4. The Interchange and Dynamic Programming Heuristics

In this section we analyze the worst-case behavior of the interchange heuristic and the dynamic programming heuristic of Baker [1]. We construct a family of problems

that gives a relative error of $(K-1)/(2K-1)$ for both of these heuristics. We prove, for the interchange heuristic, that this is the worst possible error for the subfamily of $\mathcal{P}$ with $d=0$. Thus, on the basis of worst-case analysis, neither of these methods performs as well as the greedy heuristic. Furthermore, the greedy heuristic has the smallest running time. In fact, the interchange heuristic is not known to have a polynomial bound on its running time.

It will be convenient throughout this section to treat the subfamily of $\mathcal{P}$ with $d=0$. Since in this subfamily (P) always has an optimal solution that uses K locations, the interchange heuristic will take a particularly simple form. The heuristic is initialized with an arbitrary set $J^0 \subset J$ of cardinality K. With respect to J^0 optimal values for the x_{ij} are chosen in the obvious manner mentioned in the introduction. We then determine if the solution can be improved by augmenting J^0 by a location not in J^0 and deleting from J^0 one of its present members. The procedure continues in this way until no such interchange yields an improvement. Note that the particular entering-leaving pair can be selected in a variety of ways, e.g. first improvement or maximum improvement. The worst-case analysis is independent of the method used.

Theorems 5 and 6 characterize the relative error of this heuristic. For problem (P) let z_I be the value of the solution produced by the interchange heuristic and $G_I = (z - z_I)/(z - z_R)$.

THEOREM 5. *For all* $P \in \mathcal{P}_K$, $G_I \leqslant (K-1)/(2K-1)$.

PROOF. Let $J^* \subseteq J$ denote the set of locations selected by the interchange heuristic. Define $\rho_1 \geqslant \rho_2 \geqslant \cdots \geqslant \rho_K$ by applying the greedy heuristic to (P) with J replaced by J^*. Since $d=0$, $z_R = \sum_{i \in I} u_i^1$. We note that $z_I = z_R + \rho_1 + \cdots + \rho_K$ and for any $j \notin J^*$, when j is interchanged for j_K, the objective value is $z_R + \rho_1 + \cdots + \rho_{K-1} + \rho_j(u^K)$. Since J^* cannot be improved by interchanges $\rho_K \geqslant \rho_j(u^K)$, $j \in J$. Now using $\sum_{i=1}^{m} u_i^K = z_R + \rho_1 + \cdots + \rho_{K-1}$, and $\rho_1 + \cdots + \rho_K \geqslant K\rho_K$, we have from (7)

$$\begin{aligned} z &\leqslant z_D(u^K) \leqslant z_R + \rho_1 + \cdots + \rho_{K-1} + K\rho_K \\ &\leqslant z_I + (z_I - z_R)(K-1)/K, \end{aligned}$$

which can be rewritten as $(z - z_I)/(z - z_R) \leqslant (K-1)/(2K-1)$. Q.E.D.

THEOREM 6. *Let* $P \in \mathcal{P}_K$, $K = 1, 2, \ldots,$ *be defined by* $m = 2K-1$, $n = 2K$ *and*

$$C^K = \begin{bmatrix} 1 & & & & & 1 \\ & 1 & & & & 1 \\ & & \ddots & & 0 & \vdots \\ & & & \ddots & & 1 \\ & & & & & 0 \\ & & 0 & & \ddots & 0 \\ & & & & & \vdots \\ & & & & 1 & 0 \end{bmatrix}.$$

(*The first* $2K-1$ *columns of* C^K *are unit vectors and the last column has* K *ones.*) *Then* $G_I = (K-1)/(2K-1)$, $K = 1, 2, \ldots$.

PROOF. The first K columns are an interchange solution since if any column j, $K+1 \leqslant j \leqslant 2K$, is interchanged for one of the first K columns, the increase in the objective function is 0. This gives $z_I = K$. The last K columns are an optimal set, so $z = 2K-1$. Since $z_R = 0$, $G_I = (2K-1-K)/(2K-1) = (K-1)/(2K-1)$. Q.E.D.

Since the interchange heuristic can begin with an arbitrary set of locations of cardinality K, we might choose an initial solution by applying the greedy heuristic. We will call the method that begins with the greedy solution and then applies the interchange heuristic the "greedy-interchange" heuristic. Let z_{gI} be the value of the solution produced by the greedy-interchange heuristic and $G_{gI} = (z - z_{gI})/(z - z_R)$. The family of worst-case problems used in Theorem 3 show that we can have $z_{gI} > z_g$. For example, for $K = 2$, $d = 0$ and

$$C = \begin{pmatrix} 1 & 2 & 0 \\ 1 & 0 & 2 \end{pmatrix}$$

the greedy heuristic can select locations 1 and 2. However, when the interchange heuristic is applied to this solution, location 3 replaces location 1 and we obtain the optimal solution. Nevertheless, in the worst case, the greedy-interchange heuristic is no better than the greedy heuristic. There is a family of problems, somewhat more complicated than the family used in Theorem 3, for which $G_g = [(K-1)/K]^K$ and no improvements can be made by applying the interchange heuristic.

THEOREM 7. *Let $P \in \mathcal{P}_K$, $K = 2, 3, \ldots,$ be defined by $m = K^2$, $n = 2K$ and the matrix C^K, where for $1 \leqslant j \leqslant K$*

$$\begin{aligned} c_{ij}^K &= (K-1)^{j-1}K^{K-j}, \qquad i = (j-1)K+1, \ldots, jK, \\ &= 0, \quad \textit{otherwise}; \\ c_{i,\,K+j}^K &= K^{K-1}, \qquad i = lK + j, \quad l = 0, \ldots, K-1, \\ &= 0, \quad \textit{otherwise}. \end{aligned}$$

Then $G_{gI} = [(K-1)/K]^K$, $K = 2, 3, \ldots$. (Figure 3 shows C^2 and C^3.)

$$C^2 = \begin{bmatrix} 2 & 0 & 2 & 0 \\ 2 & 0 & 0 & 2 \\ 0 & 1 & 2 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix}, \qquad C^3 = \begin{bmatrix} 9 & 0 & 0 & 9 & 0 & 0 \\ 9 & 0 & 0 & 0 & 9 & 0 \\ 9 & 0 & 0 & 0 & 0 & 9 \\ 0 & 6 & 0 & 9 & 0 & 0 \\ 0 & 6 & 0 & 0 & 9 & 0 \\ 0 & 6 & 0 & 0 & 0 & 9 \\ 0 & 0 & 4 & 9 & 0 & 0 \\ 0 & 0 & 4 & 0 & 9 & 0 \\ 0 & 0 & 4 & 0 & 0 & 9 \end{bmatrix}.$$

FIGURE 3. Examples for Theorem 7.

PROOF. For this family, proceeding as in the analysis of the family used in Theorem 3, we can show that the greedy heuristic can select the first K locations, the optimal solution contains the last K locations and $G_g = [(K-1)/K]^K$.

Now consider the interchange of location r, $K+1 \leqslant r \leqslant 2K$, and location s, $1 \leqslant s \leqslant K$. Including r improves the solution value by

$$\Delta = K^K - \sum_{j=1}^{K} (K-1)^{j-1}K^{K-j} = K^K\left[1 - K^{-1}\sum_{j=1}^{K}\alpha^{j-1}\right], \quad (\alpha = (K-1)/K)$$

$$= K^K\left[1 - K^{-1}(1-\alpha^K)/(1-\alpha)\right] = (K-1)^K.$$

Deleting location s decreases the solution value by

$$\delta_s = (K-1)\left[(K-1)^{s-1}K^{K-s}\right] = (K-1)^s K^{K-s} \geqslant (K-1)^K = \Delta.$$

Therefore no interchange yields an improvement and $G_{gI} = G_g$. Q.E.D.

We now consider Baker's dynamic programming heuristic. Problem (P) can be formulated as a dynamic program [5], but with a state-space whose cardinality grows exponentially with n. Baker approximates this state-space by a smaller one whose cardinality is linear in K and n. This approximation depends on an ordering of the elements of J and different orders can produce different solutions.

Let $J_j(r)$ be a solution set for problem (P) with $K = r$ and $J = \{1, \ldots, j\}$ and let $f(J_j(r)) = \sum_{i \in I} \max_{l \in J_j(r)} c_{il}$ be the value of an optimal solution that uses the locations in $J_j(r)$. The heuristic recursively determines values for the sets $J_j(r)$, $r = 1, \ldots, \min(j, K)$, $j = 1, \ldots, n$, which may not be optimal, and then chooses the best solution from among the sets $J_n(r)$, $r = 1, \ldots, K$. It is initialized with $J_1(1) = \{1\}$ and $f(J_1(1)) = \sum_{i \in I} c_{i1}$. Then for $r = 1, \ldots, \min(j, K)$ and $j = 2, \ldots, n$, we set

$$f(J_j(r)) = \max\left[f(J_{j-1}(r)),\quad f(J_{j-1}(r-1) \cup \{j\}) \right]$$

and

$$\begin{aligned} J_j(r) &= J_{j-1}(r) && \text{if } f(J_{j-1}(r)) \geqslant f(J_{j-1}(r-1) \cup \{j\}), \\ &= J_{j-1}(r-1) \cup \{j\} && \text{otherwise,} \end{aligned}$$

where $f(J_{j-1}(j)) = 0$ and $J_j(0) = \emptyset$ for all j. The heuristic solution is given by $z_{\mathrm{DP}} = f(J_n(r^*)) = \max_{1 \leqslant r \leqslant K} f(J_n(r))$. Note that r^* is not necessarily equal to K since it is possible to have $f(J_j(r+1)) < f(J_j(r))$, even when all of the problem data are nonnegative. (Baker appears to have missed this point since he chooses $J_n(K)$ as the solution. Our analysis assumes that the heuristic chooses the (possibly better) solution $J_n(r^*)$.)

As an example, let (P) be defined by $K = 2$ and

$$C = \begin{bmatrix} 3 & 2 & 0 \\ 0 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 0 & 2 \end{bmatrix}$$

We have

$$\begin{aligned} f(J_1(1)) &= 5, & J_1(1) &= \{1\}, \\ f(J_2(1)) &= \max[5, 4] = 5, & J_2(1) &= \{1\}, \\ f(J_2(2)) &= \max[0, 7] = 7, & J_2(2) &= \{1, 2\}, \\ f(J_3(1)) &= \max[5, 4] = 5, & J_3(1) &= \{1\}, \\ f(J_3(2)) &= \max[7, 7] = 7, & J_3(2) &= \{1, 2\}. \end{aligned}$$

Hence the heuristic solution is given by $\{1, 2\}$ and $z_{\mathrm{DP}} = 7$. The optimal solution, however, is given by the locations $\{2, 3\}$ and $z = 8$.

The following theorem shows that, with an arbitrary ordering of J, the dynamic programming heuristic is in the worst case inferior to the greedy heuristic and no better than the interchange heuristic.

THEOREM 8. *Let* $P \in \mathcal{P}_K$, $K = 1, 2, \ldots,$ *be defined as in Theorem* 6. *Then* $G_{\mathrm{DP}} = (K-1)/(2K-1)$, $K = 1, 2, \ldots$.

PROOF. Clearly $z_R = 0$. The optimal solution is given by locations $\{K+1, \ldots, 2K\}$ so that $z = 2K - 1$.

For $j = 1, \ldots, 2K-1$ and $r = 1, \ldots, \min(j, K)$ we have $J_j(r) = \{1, 2, \ldots, r\}$ and $f(J_j(r)) = r$. We then obtain $f(J_{2K}(r)) = K$, $r = 1, \ldots, K$, so that $z_{\mathrm{DP}} = \mathrm{K}$ and $G_{\mathrm{DP}} = (K-1)/(2K-1)$. Q.E.D.

Since the execution of the dynamic programming heuristic depends on the ordering of the locations, one might consider first applying the greedy heuristic and then ordering the locations so that the greedy heuristic chooses locations $\{1, \ldots, K\}$ in the natural order. But then the dynamic programming heuristic can do no better than the greedy heuristic since

THEOREM 9. *Let J be ordered so that the greedy heuristic chooses locations $\{1, \ldots, K\}$ in the natural order. Then the dynamic programming heuristic also chooses locations $\{1, \ldots, K\}$.*

PROOF. The first location chosen by the greedy heuristic (location 1) satisfies $\sum_{i\in I} c_{i1} = \max_{j\in J} \sum_{i\in I} c_{ij}$, which implies that $J_j(1) = \{1\}$, $j = 1, \ldots, n$. Now suppose that $J_j(r) = \{1, \ldots, r\}$ for all $j \geq r$, where $r < K$. At iteration $r+1$ the greedy heuristic, by hypothesis, selects location $r+1$. But this fact implies that the value of $\{1, \ldots, r, r+1\}$ is at least as great as the value of $\{1, \ldots, r, j\}$ for all $j > r+1$. Hence $J_j(r+1) = \{1, \ldots, r+1\}$, for all $j \geq r+1$ and $r < K$, which establishes that the dynamic programming solution is given by $J_n(K) = \{1, \ldots, K\}$. Q.E.D.

Baker suggests ordering J by $\sum_{i\in I} c_{ij} > \sum_{i\in I} c_{il}$ implies $j < l$. Using Theorem 9, we can prove that in the worst case the dynamic programming heuristic with Baker's proposed order can do no better than the greedy heuristic. This result can be obtained by displaying a family of problems $P \in \mathcal{P}_K$, $K = 1, 2, \ldots$, defined by the matrices C^K with the following properties: (a) the greedy heuristic chooses locations $\{1, \ldots, K\}$ in the natural order, (b) $G_g = [(K-1)/K)]^K$ and (c) the matrix C^K has equal column sums so that the locations can be put in natural order. The simplest family we have found with these properties has $m = K^{K-1}$ and $n = 2K-1$ and the general matrix C^K is rather difficult to describe. Figure 4 shows C^2 and C^3. We leave it to the reader to show that these matrices satisfy the above properties.

$$C^2 = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 0 & 2 \end{bmatrix}, \qquad C^3 = \begin{bmatrix} 1 & 3 & 7 & 0 & 0 \\ 1 & 3 & 0 & 7 & 0 \\ 1 & 3 & 0 & 0 & 7 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

FIGURE 4

5. Computational Experiments

Using several problems from the literature and real applications we have tested a number of computational ideas that are suggested by our theoretical results. The general procedure we have in mind for solving a particular problem involves three phases. In phase 1 a heuristic is applied to compute a feasible solution and possibly an upper bound on the optimal value. In phase 2 a descent method is applied to the Lagrangian dual to obtain an improved feasible solution and upper bound. Finally, to be assured of obtaining an optimal solution a branch-and-bound phase is required. This phase would use the feasible solutions and bounds from the first two phases. We have implemented and tested several options for phases 1 and 2. Although we have not implemented a branch-and-bound algorithm, the best options for phases 1 and 2 have found optimal solutions to almost every problem that we have attempted. Four heuristics were tested for phase 1: greedy, interchange beginning with an essentially random solution, interchange beginning with the feasible solution determined by the greedy heuristic (greedy-interchange) and the dynamic programming heuristic of Baker [1].

Our implementation of the interchange heuristic parallels that of Teitz and Bart [51]. The location added to the solution at each iteration is the first one that improves the solution. On the other hand, the selection of which location to remove is made to maximize the improvement in the objective value.

For phase 2 the subgradient method given in Held, Wolfe, and Crowder [22] was used. This is an iterative procedure that begins with a specified multiplier value u^0 and generates a sequence of multiplier values $\{u^k\}$. On iteration k the Lagrangian problem is solved with multipliers u^{k-1} and a new value u^k is determined from u^{k-1} and the Lagrangian solution, using the rule in [22] and the greedy solution value as a "target." At each iteration an upper bound on the optimal value is available. Additionally, the locations selected in the Lagrangian solution may be used to determine a feasible solution in the obvious manner described just after the statement of problem (P) in the introduction. A record is maintained of the best such feasible solution, and the method is terminated either when the upper bound equals the best feasible value or an iteration limit is reached.

Three alternative options were tested for specifying the initial multiplier value u^0:

(a) $u^0 = 0$,

(b) $u_i^0 = \max_j c_{ij}$, $i = 1, \ldots, m$,

(c) $u_i^0 = \max_{j \in J^*} c_{ij}$, $i = 1, \ldots, m$,

where J^* denotes the solution determined by one of the heuristics. The choice (a) was definitely the worst, while (b) proved to be slightly better than (c). Option (b) is used in the results to be reported here.

The various alternatives for phases 1 and 2 were tested on six problems. The first 4 are K-median problems taken from the literature. All of these are minimization problems having $d = 0$ and $m = n$. Our maximization procedures were applied to these problems by simply reversing the sign of C. The values of n for the 4 problems are 30, 33, 57, and 100. See [9], [26], [26], and [32] respectively for a description of the entries in C. Note that for the 30-city problem of [9] we used road distances instead of airline distances. For the problem with $n = 100$, the distances were computed from Table II [32] and were rounded to their integer parts. The fifth problem was transmitted to us by a large U.S. bank and is a real instance of the cash disbursement problem whose description begins this paper. The pertinent properties of this problem are $m = 75$, $n = 164$, $d = 0$, and $C \geqslant 0$. The last problem (also received from a bank) is a lock-box minimization problem with $m = 737$, $n = 25$, $d = 0$ and $C \geqslant 0$. In all of these problems the basic data were used with a number of different values for K.

The numerical results of our computational work are organized into four tables. Tables 1 and 2 give the objective values achieved and computation time required for the four heuristics that were tested. We also give in Table 1 either the value of an optimal solution or our best super-optimal bound on this value (obtained with the subgradient method). All objective values for the four K-median problems and the 737×25 lock-box problem are in terms of the original minimization objective while the values for the 75×164 cash disbursement problem are in terms of a maximization objective. The values of z_R for the six problems ($z_R = \sum_{i \in I} \max_j c_{ij}$ for minimization problems) are 7208539; 79336; 140054; 329501; 10200.6; and 13521.

As Table 1 indicates, the dynamic programming heuristic never found a better solution than any of the other heuristics. Differences among the other heuristics were less but greedy-interchange definitely performed the best, with interchange a close second. The computation time of greedy was about half that of interchange. The time for greedy-interchange was generally a little less than the greedy time plus the interchange time. The most attractive heuristic would appear to be greedy-interchange, given its good empirical and worst-case performance and its reasonable computational requirements.

Tables 3 and 4 report the results of our subgradient method computations. The

TABLE 1
Heuristic Objective Values

$m \times n$	K	Optimal Value	Greedy	Interchange	Greedy–Interchange	Dynamic Programming
30 × 30	2	330139	347480	330139	330139	347480
	6	115381	118469	115381	115381	143259
	10	69466	72316	69466	69466	102046
	14	38978	41636	38978	38978	56724
	18	20693	21806	20693	20693	31484
	22	8786	9899	8786	8786	11935
	26	2705	2742	2705	2705	4114
	28	1012	1012	1012	1012	1221
33 × 33	2	17474	19196	17474	17474	19196
	4	12363	12696	12609	12363	12726
	8	7461*	7953	7472	7495	8599
	17	3658	3848	3727	3676	4257
57 × 57	4	20136	20559	20559	20559	22809
	29	3342	3451	3386	3433	4347
	55	71	71	71	71	95
100 × 100	2	77659	93642	77659	77659	93642
	4	55842	59940	57723	56059	60850
	8	35364	38805	35364	35364	41198
	12	26059	28050	26059	26059	29487
	16	20036	22615	20418	20036	25540
	20	16265	18270	16444	16265	22724
†75 × 164	2	27149	27149		27149	
	3	27292*	27272		27272	
	4	27362*	27338		27338	
	5	27401*	27391		27391	
	6	27406	27406		27406	
	7	27408	27408		27408	
	8	27408	27408		27408	
737 × 25	2	8239*	8270		8270	
	3	7815*	7912		7912	
	4	7553*	7635		7635	
	5	7338*	7414		7414	
	6	7085*	7215		7215	
	7	6958*	7048		7048	
	8	6795*	6895		6895	

* This number is a super-optimal bound on the optimal value.
† Indicates maximization problem; the other ones are minimization problems.

subgradient method as described previously was applied to each problem using $u_i^0 = \max_j c_{ij}$ for all i. The method was run until a bound was discovered which equaled the best feasible solution obtained thus far (in which case the problem has been solved optimally and therefore the LP solution equals the integer solution) or until 150 iterations had been performed. In eleven cases the upper limit of 150 iterations was reached before optimality. For these cases the best Lagrangian bound and best feasible value obtained are given in Table 4. Note that the times shown for the subgradient method do not include the time required to initialize the target using the greedy heuristic.

We have also displayed solution times taken from [37] and [47] for the best previously existing algorithms. The subgradient method clearly out-performed Khumawala's algorithm. Comparison with Schrage's algorithm is harder because his times allow for input and output of the problem. However, even making a generous allowance for I/O the subgradient times appear to be somewhat less than Schrage's.

TABLE 2

Heuristic Computation Times—370-168 Seconds with I/O Time not Included

$m \times n$	K	Greedy Time	Interchange Time	Interchange No. of Interchanges	Greedy-Interchange Time	Greedy-Interchange No. of Interchanges	Dynamic Programming Time
30 × 30	2	0.001	0.005	2	0.007	3	0.003
	6	0.003	0.001	4	0.027	1	0.003
	10	0.017	0.039	4	0.040	4	0.013
	14	0.023	0.020	7	0.050	2	0.033
	18	0.033	0.050	6	0.073	1	0.043
	22	0.040	0.063	4	0.110	1	0.053
	26	0.046	0.063	2	0.116	1	0.050
	28	0.056	0.067	2	0.116	0	0.053
33 × 33	2	0.003	0.010	14	0.010	2	0.003
	4	0.003	0.020	18	0.013	2	0.010
	8	0.017	0.037	17	0.070	5	0.023
	17	0.040	0.056	13	0.120	5	0.037
57 × 47	4	0.027	0.052	10	0.063	0	0.043
	29	0.203	0.296	20	0.456	2	0.223
	55	0.453	0.369	2	0.809	0	0.253
100 × 100	2	0.032	0.110	8	0.121	8	
	4	0.088	0.240	13	0.291	10	
	8	0.161	0.369	13	0.567	13	
	12	0.271	0.679	25	0.930	14	
	16	0.331	0.619	21	0.970	15	
	20	0.418	0.805	30	1.362	19	
†75 × 164	2	0.156			0.276	0	
	3	0.250			0.445	0	
	4	0.359			0.828	0	
	5	0.419			0.935	0	
	6	0.485			0.828	0	
	7	0.589			1.330	0	
	8	0.675			1.461	0	
737 × 25	2	0.230			0.400	0	
	3	0.333			0.581	0	
	4	0.460			0.780	0	
	5	0.542			1.031	0	
	6	0.666			1.139	0	
	7	0.795			1.371	0	
	8	0.918			1.513	0	

† Indicates maximization problem; the other ones are minimization problems.

TABLE 3

Subgradient Experiments

$m \times n$	K	Subgradient Method Iterations	Subgradient Method Time on 370-168 not including I/O	Khumawala Algorithm Time on 370-165 not including I/O
30 × 30	2	29	0.2	2.00
	6	28	0.2	28.10
	10	66	0.6	7.41
	14	91	0.8	15.68
	18	34	0.4	3.21
	22	50	0.5	0.65
	26	43	0.5	0.54
	28	60	0.8	0.40

TABLE 4
Subgradient Experiments

		Subgradient Method				Schrage Algorithm	
$m \times n$	K	Iterations	Time on 370-168 not including I/O	Best Lagrangian Bound Obtained	Best Feasible Value Obtained	Pivots	Time on 370-165 including I/O
33 × 33	2	10	0.1			119	1.39
	4	58	0.4			167	2.26
	8	150	1.5	7461	7472	95	2.17
	17	29	0.4			77	1.12
57 × 57	4	41	1.1			439	10.58
	29	81	2.2			100	3.97
	55	10	0.5			58	2.81
100 × 100	2	25	1.3				
	4	64	3.4				
	8	84	4.8				
	12	118	7.0				
	16	116	7.2				
	20	94	6.3				
†75 × 164	2	48	3.3				
	3	150	10.3	27292	27272		
	4	150	10.5	27362	27338		
	5	150	10.5	27401	27391		
	6	2	0.7				
737 × 25	2	150	15.0	8239	8270		
	3	150	16.3	7815	7912		
	4	150	17.6	7553	7635		
	5	150	18.9	7338	7414		
	6	150	20.1	7085	7215		
	7	150	21.5	6958	7048		
	8	150	23.0	6795	6895		

† Indicates maximization problem; the other ones are minimization problems.

References

1. BAKER, K. R., "A Heuristic Approach to Locating a Fixed Number of Facilities," *Logistics and Transportation Review* (Autumn 1974).
2. COOK, S. A., "The Complexity of Theorem-proving Procedures," *Proc. 3rd. Annual ACM Symp. on the Theory of Computing* (1971), pp. 151–158.
3. COOPER, L., "Location-Allocation Problems," *Opns. Res.*, Vol. 11 (1963), pp. 331–343.
4. ———, "Heuristic Methods for Location-Allocation Problems," *SIAM Rev.*, Vol. 6 (1964), pp. 37–53.
5. CURRY, G. AND SKEITH, R., "A Dynamic Programming Algorithm for Facility Location and Allocation," *AIIE Trans.*, Vol. 1 (1969), pp. 133–138.
6. DIEHR, G., "An Algorithm for the p-Median Problem," Paper No. 191, Western Management Science Institute, UCLA (June 1972).
7. DRYSDALE, J. K. AND SANDIFORD, P. J., "Heuristic Warehouse Location—A Case History Using a New Method," *CORS*, Vol. 7 (1969), pp. 45–61.
8. EFFROYMSON, M. A. AND RAY, T. L., "A Branch-Bound Algorithm for Plant Location," *Opns. Res.*, Vol. 14 (1966), pp. 361–368.
9. EL-SHAIEB, A. M., "A New Algorithm for Locating Sources Among Destinations," *Man. Sci.*, Vol. 20 (1973), pp. 221–231.
10. EVERETT, H. M., "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," *Opns. Res.*, Vol. 11 (1963), pp. 399–417.
11. FELDMAN, E., LEHRER, F. A. AND RAY, T. L., "Warehouse Location Under Continuous Economies of Scale," *Man. Sci.*, Vol. 12 (1966), pp. 670–684.
12. FISHER, M. L., "Optimal Solution of Scheduling Problems Using Lagrange Multipliers," *Opns. Res.*, Vol. 21 (1973), pp. 1114–1127.
13. ———, "A Dual Algorithm for the One-Machine Scheduling Problem," Report No. 243, Dept. of Opns. Res., Cornell University (to appear in *Math. Prog.*).
14. ——— AND SHAPIRO, J. F., "Constructive Duality in Integer Programming," *SIAM J. on Applied Math.*, Vol. 27 (1974), pp. 31–53.

15. FRANCIS, R. M. AND GOLDSTEIN, J. M., "Location Theory: A Selective Bibliography," *Opns. Res.*, Vol. 22 (1974), pp. 400–409.
16. GAREY, M. R. AND GRAHAM, R. L., "Bounds for Multiprocessor Scheduling with Resource Constraints," *SIAM J. on Computing*, Vol. 4 (1975), pp. 187–200.
17. GARFINKEL, R. S., NEEBE, A. W. AND RAO, M. R., "An Algorithm for the *M*-Median Plant Location Problem," *Transp. Sci.*, Vol. 8 (1974), pp. 217–236.
18. GEOFFRION, A. M., "Lagrangian Relaxation for Integer Programming," *Math. Prog. Study*, Vol. 2 (1974), pp. 82–114.
19. ———, "A Guide to Computer-Assisted Methods for Distribution Systems Planning," *Sloan Management Review*, Vol. 16 (1975), pp. 17–41.
20. HELD, M. AND KARP, R. M., "The Traveling Salesman Problem and Minimum Spanning Trees," *Opns. Res.*, Vol. 18 (1970), pp. 1138–1162.
21. ——— AND ———, "The Traveling Salesman Problem and Minimum Spanning Trees: Part II," *Math. Prog.*, Vol. 1 (1971), pp. 6–25.
22. ———, WOLFE, P. AND CROWDER, H. P., "Validation of Subgradient Optimization," *Math. Prog.*, Vol. 6 (1974), pp. 62–88.
23. JARVINEN, P., RAJALA, J. AND SINERRO, J., "A Branch-and-Bound Algorithm for Seeking the *p*-Median," *Opns. Res.*, Vol. 20 (1972), pp. 173–178.
24. JOHNSON, D. S., "Approximation Algorithms for Combinatorial Problems," *J. of Computer and Systems Sciences*, Vol. 9 (1974), pp. 256–278.
25. ———, DEEMERS, A., ULLMAN, J. D., GAREY, M. R. AND GRAHAM, R. L., "Worst-Case Performance Bounds for Simple One-Dimensional Packing Algorithms," *SIAM J. on Computing*, Vol. 3 (1974), pp. 299–326.
26. KARG, R. L. AND THOMPSON, G. L., "A Heuristic Approach to Solving Traveling Salesman Problems," *Man. Sci.*, Vol. 10 (1964), pp. 225–248.
27. KARP, R. M., "Reducibility Among Combinatorial Problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, eds., Plenum Press, New York, 1972, pp. 85–104.
28. ———, "On the Computational Complexity of Combinatorial Problems," *Networks*, Vol. 5 (1975), pp. 45–68.
29. KHUMAWALA, B. M., "An Efficient Branch-and-Bound Algorithm for the Warehouse Location Problem," *Man. Sci.*, Vol. 18 (1972), pp. B-718–731.
30. ———, NEEBE, A. W. AND DANNENBRING, D. G., "A Note on El-Shaieb's New Algorithm for Locating Sources Among Destinations," *Man. Sci.*, Vol. 21 (1974), pp. 230–233.
31. KRAUS, A., JANSSEN, C. AND MCADAMS, A., "The Lock-box Location Problem, a Class of Fixed Charge Transportation Problems," *J. of Bank Res.*, Vol. 1 (1970), pp. 51–58.
32. KROLAK, P., FELTS, W. AND MARBLE, G., "A Man-Machine Approach towards Solving the Traveling Salesman Problem," *CACM*, Vol. 14 (1971), pp. 327–334.
33. KUEHN, A. A. AND HAMBURGER, M. J., "A Heuristic Program for Locating Warehouses," *Man. Sci.*, Vol. 9 (1963), pp. 643–666.
34. LEVY, F. K., "An Application of Heuristic Problem Solving to Accounts Receivable Management," *Man. Sci.*, Vol. 12 (1966), pp. B-236–244.
35. LORIE, J. AND SAVAGE, L. J., "Three Problems in Capital Rationing," *J. of Bus.*, Vol. 28 (1955), pp. 229–239.
36. MAIER, S. F. AND VANDER WEIDE, J. H., "The Lock-Box Location Problem: A Practical Reformulation," Grad. School of Bus. Adm. Paper No. 87, Duke University (1973).
37. "Making Millions by Stretching the Float," *Businessweek* (November 23, 1974), pp. 88–90.
38. MANNE, A. S., "Plant Location under Economies-of-Scale-Decentralization and Computation," *Man. Sci.*, Vol. 11 (1964), pp. 213–235.
39. MARANZANA, F. E., "On the Location of Supply Points to Minimize Transport Costs," *Opnal. Res. Quart.*, Vol. 15 (1964), pp. 261–270.
40. MARSTEN, R. E., "An Algorithm for Finding Almost All the Medians of a Network," Center for Math Studies in Economics and Man. Sci. Paper No. 23, Northwestern University (1972).
41. NEMHAUSER, G. L. AND TROTTER, L. E., JR., "Vertex Packings: Structural Properties and Algorithms," *Math. Prog.*, Vol. 8 (1975), pp. 232–248.
42. REITER, S. AND SHERMAN, G., "Discrete Optimizing," *SIAM J.*, Vol. 13 (1965), pp. 864–899.
43. ROSENKRANTZ, D. J., STEARNS, R. E. AND LEWIS, P. M., "Approximation Algorithms for the Traveling Salesperson Problem," *Proc. of the 15th IEEE Symp. on Switching and Automata Theory*, (1974), pp. 33–42.
44. ROSS, G. T. AND SOLAND, R. M., "A Branch and Bound Algorithm for the Generalized Assignment Problem," *Math. Prog.*, Vol. 8 (1975), pp. 91–103.
45. SAHNI, S., "Approximate Algorithms for the 0/1 Knapsack Problem," *JACM*, Vol. 22 (1975), pp. 115–124.
46. ——— AND GONZALES, T., "P-Complete Approximation Problems," *JACM*, Vol. 23 (1976), pp. 555–565.

47. SCHRAGE, L., "Implicit Representation of Variable Upper Bounds in Linear Programming," *Math. Prog. Studies*, Vol. 4 (1975), pp. 118–132.
48. SHAPIRO, J. F., "Generalized Lagrange Multipliers in Integer Programming," *Opns. Res.*, Vol. 19 (1971), pp. 68–76.
49. SPIELBERG, K., "Algorithms for the Simple Plant-Location Problem with Some Side Conditions," *Opns. Res.*, Vol. 17 (1969), pp. 85–111.
50. ———, "Plant Location with Generalized Search Origin," *Man. Sci.*, Vol. 16 (1969), pp. 165–178.
51. TEITZ, M. B. AND BART, P., "Heuristic Methods for Estimating the Generalized Vertex Median of a Weighted Graph," *Opns. Res.*, Vol. 16 (1968), pp. 955–961.