ELSEVIER

European Journal of Operational Research 88 (1996) 114–123

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

A Lagrangean heuristic for the maximal covering location problem

Roberto Diéguez Galvão [a,*], Charles ReVelle [b]

[a] *COPPE, Federal University of Rio de Janeiro, Rio de Janeiro, Brazil*

[b] *Deparment of Geography and Environmental Engineering, The Johns Hopkins University, 313 Ames Hall, Baltimore, MD 21218-2686, USA*

Received February 1993

Abstract

We develop a Lagrangean heuristic for the maximal covering location problem. Upper bounds are given by a vertex addition and substitution heuristic and lower bounds are produced through a subgradient optimization algorithm. The procedure was tested in networks of up to 150 vertices. A duality gap was generally present at the end of the heuristic for the larger problems. The test problems were run in an IBM 3090-600J 'super-computer'; the maximum computing time was kept below three minutes of CPU.

Keywords: Maximal covering location problem; Lagrangean relaxation; Heuristics

1. Introduction

The maximal covering location problem (MCLP) was one of the first location covering models studied in the vast literature that presently exists on models that provide 'coverage' to demand areas. A demand area is said to be covered by a facility if it is within a required distance or time from the facility.

The simplest of these covering models is the location set covering problem (LSCP), which seeks to position the minimum number of facilities that are necessary to cover all demand areas within S distance or time units. A related problem is the p-center problem (PCP), which seeks the location of p facilities such that the maximum distance (time) from any demand area to its nearest facility is minimized.

Location covering models generally address the location of urban public facilities. Both LSCP and PCP require that all demand areas be covered, and this may require excessive resources not always available to the public authorities. MCLP does not require that all demand areas be covered; in this case the objective is to locate p facilities such that the maximal possible population is covered within distance (time) S. Church and ReVelle [2] were the first authors to study MCLP. White and Case [14] worked in a similar problem, which sought to locate p facilities that could cover the maximum number of demand areas (rather than the maximal population).

Church and ReVelle [2] developed two heuristic procedures to solve MCLP, namely the Greedy Adding and the Greedy Adding with Substitution algorithms. These are similar to vertex addition

* Corresponding author. Visiting Professor at The Johns Hopkins University, 1991/92.

SSDI 0377-2217(94)00159-A

and substitution algorithms developed for other location problems such as the simple plant location problem and the p-median problem (see, for example, [4], [11] and [13]), and do not guarantee global optimality. The authors also applied linear programming (LP) to solve MCLP, but in this case optimality is guaranteed only if the LP solutions are all-integer (zero–one). Their computational experience with 30- and 55-vertex networks using several values of p and S was that approximately 80 percent of the LP solutions were zero-one. Fractional LP solutions were resolved by inspection.

When we focus on the methods used to solve location covering problems to optimality, we discover that they usually resort to the use of the LP relaxation of the respective 0–1 formulations. The mathematical formulations used are such that the corresponding LP relaxations produce integer solutions in the majority of the cases. When fractional LP solutions occur, a branch-and-bound algorithm is generally used to produce integer solutions. The computational experience reported on these problems is however restricted to medium-sized problems, not larger than (55 demand areas) $\times$ (55 potential facility sites).

The use of Lagrangean Relaxation (LR) to solve location covering problems has been limited to very few instances. Weaver and Church [12] experimented both with the p-median equivalent of MCLP and Lagrangean bounding procedures for MCLP, with inconclusive results. A recent review of the links between the p-median, the generalized assignment and the maximal covering location problems was made by Pastor and Alminana [8], who also analysed some exact and heuristic methods to solve these problems and gave an overview of the most relevant extensions and applications of MCLP. Pirkul and Schilling [9] used LR to solve a capacitated MCLP with backup coverage. Church et al. [3] solved a bicriterion MCLP as a p-median problem that uses LR as a solution technique. Because LR is generally a more efficient bounding technique than LP relaxation in the solution of combinatorial optimization problems, we decided to investigate the use of LR to solve MCLP.

In the present paper we develop a Lagrangean heuristic to solve MCLP. The algorithm interacts between an upper bound, obtained through a 'greedy'-interchange heuristic, and a lower bound, obtained via a classical subgradient optimization algorithm, until a stop rule is reached. Computational results are given for problems of up to the size (150 demand areas) $\times$ (150 potential facility sites). While the smaller problems were almost always solved to optimality, a duality gap was generally present at the end of the procedure for the larger problems.

Our approach to solve MCLP using LR differs from the Weaver and Church [12] bounding procedure for the same problem in the following way. Weaver and Church [12] also used subgradient optimization to maximize the Lagrangean dual (in order to obtain a 'tight' lower bound), but did not attempt to interactively improve either the upper or the lower bound during their procedure. The Lagrangean heuristic we developed interacts between upper and lower bounds, attempting to improve both bounds in each iteration of the algorithm. This provides better bounds for the problem and therefore a better approximate solution for MCLP, especially for the larger problems. We also provide computational experience for substantially larger problems than those tested in [12].

The paper is organized as follows. In Section 2 we analyse two Lagrangean relaxations of MCLP. Section 3 describes the Lagrangean heuristic we developed for MCLP; general observations about the upper and lower bounding phases of the heuristic are followed by a step-by-step description of the algorithm and by its illustration through a simple example. Section 4 is dedicated to computational results and this is followed by the conclusions in Section 5.

2. Lagrangean relaxations of MCLP

Let the mathematical programming formulation for the maximal covering location problem be given by:

(MCLP)

$$v(\text{MCLP}) = \max\left\{ \sum_{i \in I} f_i y_i \right\} \equiv \min\left\{ - \sum_{i \in I} f_i y_i \right\} \tag{1}$$

subject to

$$y_i \leq \sum_{j \in N_i} x_j, \quad i \in I, \tag{2}$$

$$\sum_{j \in J} x_j = p, \tag{3}$$

$$y_i \in \{0, 1\}, \quad i \in I, \tag{4}$$

$$x_j \in \{0, 1\}, \quad j \in J, \tag{5}$$

where $I = \{1, \ldots, m\}$ is the set of demand areas, $J = \{1, \ldots, n\}$ is the set of potential facility sites, $N_i = \{j \in J \mid d_{ij} \leq S\}$ is the set of facility sites that are within the critical distance S from demand area i, f_i is the population of demand area i, d_{ij} is the shortest distance from demand area i to facility site j, S is the distance beyond which a demand area is considered uncovered, p is the number of facilities to be located and y_i and x_j are the decision variables. $y_i = 1$ if demand area i is covered ($y_i = 0$ otherwise); $x_j = 1$ means that a facility must be located at site $j \in J$ ($x_j = 0$ otherwise).

In the formulation above restrictions (2) state that for a demand area i to be covered ($y_i = 1$) there must be at least one facility j located within the critical distance S from i ($N_i \neq \emptyset$ for the corresponding i). Restriction (3) limits the number of facilities in the solution to p and restrictions (4)–(5) define the 0–1 nature of the decision variables.

There are two possible Lagrangean relaxations of MCLP: MCLP_u, resulting from the dualization of restrictions (2) (using dual vector $u \geq 0$), and MCLP_v, resulting from the dualization of (3) (using the dual multiplier v (scalar), unrestricted in sign). The Lagrangean problem MCLP_v is a variable upper bound (VUB) 0–1 problem whose solution is not readily available. For this reason we did not consider using relaxation MCLP_v in the Lagrangean heuristic we developed for MCLP.

Lagrangean Relaxation MCLP_u

(MCLP_u)

$$v_u(\text{MCLP}) = \min\left\{ -\sum_{i \in I} f_i y_i + \sum_{i \in I} u_i \left(y_i - \sum_{j \in N_i} x_j \right) \right\}$$

$$\equiv \min\left\{ \sum_{i \in I} (u_i - f_i) y_i - \sum_{i \in I} u_i \sum_{j \in N_i} x_j \right\} \tag{6}$$

subject to

$$\sum_{j \in J} x_j = p, \tag{7}$$

$$y_i, x_j \in \{0, 1\}, \quad i \in I, \quad j \in J. \tag{8}$$

Notice that since $u \geq 0$, $v_u(\text{MCLP}) \leq (v(\text{MCLP}))_{\min}$ is a lower bound for the minimization problem. It is also important to note that MCLP_u has the integrality property; the bounds produced by LR, therefore, will not be better than those produced by LP relaxation. Other considerations, however, justify the use of LR in this case: lower computing times for large problems and the possibility of interactively obtaining good primal solutions in the course of the Lagrangean heuristic procedure.

MCLP_u can be solved independently for the y_i's and the x_j's, for any given dual vector $u = (u_1, \ldots, u_m)$. Let (y_i^*, x_j^*) correspond to the optimal solution of MCLP_u. Then:

(a) In relation to the y_i's, simply set

$$y_i^* = \begin{cases} 1 & \text{if } u_i \leq f_i, \\ 0, & \text{otherwise.} \end{cases}$$

(b) In relation to the x_j's, MCLP_u is a knapsack problem that can be solved by inspection in the following way:

(i) order the coefficients of the x_j's in decreasing order of absolute value;

(ii) make the x_j's corresponding to p largest coefficients (absolute value) equal to 1 ($x_j^* = 1$), all other $x_j^* = 0$.

A primal solution v_{primal} can be readily obtained from the optimal solution of MCLP_u: for each of the p $x_j^* = 1$, find the demand areas that are within distance S from the corresponding facility and make the corresponding y_i's equal to one ($y_i^{\text{primal}} = 1$), all other $y_i^{\text{primal}} = 0$. Then

$$v_{\text{primal}} = \sum_{i \in I} f_i y_i^{\text{primal}}$$

is an upper bound for MCLP.

3. A Lagrangean heuristic for MCLP

In the algorithm given below we obtain lower bounds for MCLP using Lagrangean relaxation MCLP_u. Upper bounds are obtained through

'greedy'-interchange and interchange heuristics. After an initial upper bound is obtained, a subgradient optimization algorithm which maximizes the dual of MCLP_u is activated. This subgradient algorithm solves the problem

(D)

$$v_{\text{D}}(\text{MCLP}) = \max_{u \geq 0} v_u(\text{MCLP}).$$

In each iteration of the subgradient optimization algorithm the lower bound is tentatively updated. A primal solution (upper bound v_{primal}) for MCLP is also obtained using the procedure described in Section 2. This upper bound is generally a weak bound, so under certain conditions given in Section 3.2 the corresponding primal solution is used as an initial solution in an interchange heuristic that attempts to obtain a stronger upper bound. The MCLP upper bound is then tentatively updated before a new iteration of the subgradient optimization algorithm takes place.

3.1. *The lower bounding phase of the Lagrangean heuristic*

This phase corresponds to a classical subgradient optimization algorithm. Given an initial vector u^0, a sequence $\{u^k\}$ is generated by the rule

$$u_i^{k+1} = \max\{0,\ u_i^k + \theta_k \gamma_i^k\}, \quad i = 1, \ldots, m,$$

where γ^k is a subgradient at $u = u^k$ and $\theta_k > 0$ is the step size. We calculate the step size at iteration k applying the commonly used formula

$$\theta_k = \alpha_k \left(v^{*k} - v_{u^k}\right) / \| \gamma^k \|^2,$$

where α_k, the value of the step size parameter at iteration k, is a scalar satisfying $0 < \alpha_k \leq 2$; v^{*k} is the upper bound on v_{D} available at iteration k, usually obtained by applying a heuristic to solve MCLP; v_{u^k} is the lower bound at iteration k.

The subgradient vector at iteration k is given by

$$\gamma^k = \left[\gamma_i^k\right],$$

with

$$\gamma_i^k = y_i^{*k} - \sum_{j \in N_i} x_j^{*k},$$

where $(y_i^{*k},\ x_j^{*k})$ is the solution of MCLP_u at iteration k. In relation to the initial vector u^0, two strategies were tested, namely $u_i = 0,\ \forall i$, and $u_i = \text{f}_i,\ \forall i$. Although the results did not differ substantially, a faster convergence rate and better bounds were generally obtained when we used $u_i = 0,\ \forall i$. The computational results of Section 4 correspond to this strategy.

We experimented with several rules for updating the step size parameter α. The rules that performed better in relation to the convergence of the subgradient optimization procedure were the following:

(i) Always start with $\alpha = 2$;
(ii) For the 55-vertex network, halve the value of α if the value of the lower bound fails to improve after n iterations of the subgradient optimization algorithm;
(iii) For larger problems, halve the value of α if the lower bound fails to improve after $\frac{1}{4}n$ iterations of the algorithm.

The following stop rules were used to end the Lagrangean heuristic:

(i) If the current absolute value of the difference between the upper and lower bounds is less than one unit, Stop: as MCLP is an integer programming problem, a difference that is less than one unit indicates that optimality has been achieved. The optimal solution for the problem is given by the current upper bound;
(ii) If $\gamma_i = 0,\ \forall i$, Stop. An optimal solution for v_{D} has been obtained, but a duality gap may exist. The best available solution is given by the current upper bound;
(iii) If $\theta < 0.01$, Stop. A duality gap exists and the best available solution is given by the current upper bound;
(iv) If

no. of iterations > 500

for problems up to the size 100×100, or if

no. of iterations > 1000

for larger problems, Stop. The situation is the same as in condition (iii) above;
(v) If total coverage has been achieved by a primal solution, Stop. The optimal solution

for the problem is given by this primal solution.

3.2. The upper bounding phase of the heuristic

A 'greedy'-interchange heuristic is used to obtain an initial upper bound for MCLP. This heuristic consists of a vertex addition phase used in conjunction with a vertex substitution phase and is similar to heuristics developed for other facility location problems, for example the heuristic of Eilon and Galvão [4] for the p-median problem, but with the MCLP objective function used to evaluate alternative solutions. A 'stand-alone' interchange heuristic is used improve the primal solution v_{primal} whenever either (i) an improved lower bound (given by v_u(MCLP)) is obtained, or (ii) the value of the step size parameter α is halved in the subgradient optimization algorithm.

The 'stand-alone' heuristic could in fact have been used in every iteration of the subgradient optimization algorithm, but in order to keep computing times down we chose to activate it only in the instances reported above.

The 'greedy'-interchange heuristic starts by finding an optimal solution for MCLP for $p = 1$. The vertex (demand area) that produces the maximal possible increase in total population covered is then added to the solution, and the interchange heuristic is applied to this solution to produce the heuristic solution for $p = 2$. The heuristic proceeds in a stepwise fashion, with the interchange algorithm being applied to the set of cardinality p generated by the vertex addition ('greedy') algorithm. The procedure continues until either total coverage is achieved or the desired value of p is reached.

The interchange algorithm belongs to a family of heuristics based on local optimization and the idea of λ-optimality, first introduced by Lin [7] for the traveling salesman problem and subsequently extended for a variety of combinatorial optimization problems, see for example Kerningnan and Lin [6], Christofides and Eilon [1]. λ substitution procedures involve examining a given solution set $\mathcal{S}$ consisting of p points by exchanging λ of its points (where $\lambda \leq p$) with λ points taken from the set $\mathcal{R}$ of eligible points to belong to the solution. The replacement set of λ points must evidently satisfy the condition that at least one of the λ points does not belong to $\mathcal{S}$.

The number of potential substitutions that must be performed in order to ensure that a set is λ-optimal increases rapidly with λ and hence practical algorithms cannot use values of λ much above 2. When we used the 'greedy'-interchange heuristic to obtain an initial upper bound for MCLP we experimented with values of $\lambda = 1$ and $\lambda = 2$ for the interchange algorithm. The bounds obtained with $\lambda = 2$ were as expected superior to those obtained with $\lambda = 1$, but computing times became prohibitive when we used $\lambda = 2$ for the larger problems. The 'stand-alone' interchange heuristic to improve v_{primal} was always used with $\lambda = 1$.

Since λ-optimal procedures produce only local optimal solutions, the final solution depends to a great extent on the p points chosen for the initial solution set $\mathcal{S}$. Given that the interchange heuristic is used to improve v_{primal} several times in the subgradient optimization algorithm, starting each time from a different set of p points, there is a high probability that a good lower bound for MCLP will be available at the end of the Lagrangean heuristic, regardless of how 'tight' the initial lower bound provided by the 'greedy'-interchange heuristic is. For this reason, and to keep computing times down to reasonable values, we finally decided to use $\lambda = 1$ whenever the interchange heuristic was activated, either as part of the 'greedy'-interchange heuristic or as a 'stand-alone' procedure.

3.3. Steps of the Lagrangean heuristic

We are now able to describe the Lagrangean heuristic developed for MCLP. The steps of the algorithm are:

Step 1. Initialization:

Step 1A. Using the 'greedy'-interchange heuristic obtain an initial value for UB_{MCLP}, upper bound for MCLP;

Step 1B. Initialize LB_{MCLP}, lower bound for MCLP, as $LB_{\text{MCLP}} = -\infty$;

Step 1C. Choose initial values u_i^0 for the Lagrangean multipliers: $u_i^0 = 0$, $i = 1, 2, \ldots, m$;

Step 1D. Set $k = 0$, *iter_counter* $= 0$, *update_counter* $= 0$, $\alpha_k = 2$;

Step 2. Solve the Lagrangean problem MCLP_u:

Step 2A. Set *iter_counter* $\leftarrow$ *iter_counter* $+ 1$, *update_counter* $\leftarrow$ *update_counter* $+ 1$;

Step 2B. Calculate y_i^{*k}, x_j^{*k} as outlined in Section 2;

Step 2C. Compute

$$v_{u^k}(\text{MCLP}) = \sum_{i \in I} (u_i^k - f_i) y_i^{*k} - \sum_{i \in I} u_i^k \sum_{j \in N_i} x_j^{*k};$$

Step 3. Obtain a primal solution for MCLP:

Step 3A. Obtain

$$y_i^{\text{primalk}}, \; i = 1, 2, \ldots, m,$$

as outlined in Section 2;

Step 3B. Compute

$$v_{\text{primalk}} = \Sigma_{i \in I} f_i y_i^{\text{primalk}};$$

Step 4. Update the bounds:

Step 4A. If $v_{u^k}(\text{MCLP}) > LB_{\text{MCLP}}$ then:

(i) Set $v_{u^k}(\text{MCLP}) \leftarrow LB_{\text{MCLP}}$;

(ii) Apply the interchange heuristic to v_{primalk} to obtain interchange (v_{primalk}), where interchange (v_{primalk}) is the value of the solution that results from applying the interchange heuristic to v_{primalk};

Step 4B. If

$$v_{\text{primalk}} \left(\text{interchange}(v_{\text{primalk}})\right) < UB_{\text{MCLP}},$$

then set

$$v_{\text{primalk}} \left(\text{interchange}(v_{\text{primalk}})\right) \leftarrow UB_{\text{MCLP}};$$

Step 5. Update the parameters and check for termination conditions:

Step 5A. If

$$\textit{update\_counter} = \textit{update\_counter\_limit},$$

then:

(i) Set

$$\alpha_k \leftarrow \alpha_k / 2,$$

$$\textit{update\_counter} \leftarrow 0;$$

(ii) Apply the interchange heuristic to v_{primalk} to obtain the value of interchange(v_{primalk});

(iii) If

$$\text{interchange}(v_{\text{primalk}}) < UB_{\text{MCLP}},$$

set interchange$(v_{\text{primalk}}) \leftarrow UB_{\text{MCLP}}$;

Step 5B. Check for the termination conditions (see Section 3.1).

If any of the termination conditions is met then: STOP;

else: Go to Step 6;

Step 6. Update the Lagrangean multipliers:

Step 6A. Set

$$u^k \leftarrow \max\{0, u^k + \theta_k \gamma_k\};$$

Step 6B. Set $k \leftarrow k + 1$;

Step 6C. Go to Step 2.

Note that the values of both *update_counter_limit* and *iter_count_limit* depend on problem size. These values have been defined in Section 3.1.

3.4. Illustration of the Lagrangean heuristic through a simple example

We now illustrate the Lagrangean heuristic through a simple example. In this example we wish to locate 3 facilities in the 10-vertex network of Garfinkel, Neebe and Rao [5] for the p-median problem (see p.231 of their paper), in order to maximize the total population covered within a critical distance of S = 5 units. The population values we used for the vertices of this network were

$f_1 = f_2 = 8$;
$f_3 = f_4 = 6$;
$f_5 = 5$;
$f_6 = f_7 = 4$;
$f_8 = 3$;
$f_9 = 2$; and
$f_{10} = 1$.

The optimal solution was obtained after 12 iterations. Vertices 8, 9 and 10 were not covered; the total population covered was 41, corresponding to 87.23% of the total population. In Table 1 we show the values of the upper and lower bounds for each of the 12 iterations. As the upper bound obtained in Step 1A of the Lagrangean heuristic corresponds to the optimal solution of this problem, this bound could not be improved during the heuristic procedure.

Table 1
Values of lower and upper bounds in each of the 12 iterations

Iteration	Lower bound	Best lower bound	Best upper bound
1	−47.00	−47.00	−41.00
2	−45.50	−45.50	−41.00
3	−46.82	−45.50	−41.00
4	−45.44	−45.44	−41.00
5	−46.89	−45.44	−41.00
6	−44.46	−44.46	−41.00
7	−45.56	−44.46	−41.00
8	−43.53	−43.53	−41.00
9	−43.47	−43.47	−41.00
10	−43.12	−43.12	−41.00
11	−43.12	−43.12	−41.00
12	−41.53	−41.53	−41.00

4. Computational results

The Lagrangean heuristic was coded in Fortran 77 and run on an IBM 3090-600J 'supercomputer'. Two sets of data were used to test the heuristic: data from the literature and randomly generated networks.

The data from the literature corresponds to the 55-vertex network designed by Swain [10] and later used both by Church and ReVelle [2] and Weaver and Church [12]. The randomly generated networks are networks of 100 and 150 vertices which were generated as follows.

The number of arcs of the randomly generated

Table 2
Computational results – 55-vertex network [a]

n	p	S	Iter	Ubound	Lbound	% Dif	% Cov.	Ncalls	Comp. times [b]		
									T Gl Hr.	T I Hr.	Ttime
55	3	6	10	446.07	446.00	0.02	69.69	10	0.04	0.34	0.53
55	4	6	15	470.38	470.00	0.08	73.44	12	0.06	0.62	0.90
55	5	6	23	491.33	491.00	0.07	76.72	16	0.09	1.13	1.54
55	6	6	43	509.47	509.00	0.09	79.53	20	0.13	1.76	2.46
55	7	6	38	525.32	525.00	0.06	82.03	24	0.16	3.09	3.75
55	8	6	42	540.29	540.00	0.05	84.38	23	0.21	3.55	4.32
55	3	8	16	499.18	499.00	0.04	77.97	12	0.04	0.41	0.68
55	4	8	16	538.42	538.00	0.08	84.06	13	0.07	0.77	1.08
55	5	8	22	564.54	564.00	0.10	88.12	15	0.10	1.23	1.64
55	6	8	43	584.34	584.00	0.06	91.25	19	0.14	2.02	2.74
55	7	8	500	598.60	598.00	0.10	93.44	59	0.18	5.91	12.49
55	8	8	500	611.65	610.00	0.27	95.31	43	0.22	7.44	14.11
55	3	10	24	548.36	548.00	0.07	85.62	16	0.04	0.63	1.02
55	4	10	57	581.49	581.00	0.08	90.78	27	0.09	1.60	2.46
55	5	10	68	609.43	609.00	0.07	95.16	23	0.15	1.99	3.06
55	6	10	66	625.53	625.00	0.08	97.66	16	0.19	1.37	2.46
55	7	10	105	633.53	633.00	0.08	98.91	22	0.23	2.27	3.90
55	8	10	500	639.06	638.00	0.17	99.69	34	0.28	4.63	11.56
55	3	12	500	575.77	574.00	0.31	89.69	47	0.05	1.90	8.56
55	4	12	269	610.60	610.00	0.10	95.31	32	0.08	2.12	5.81
55	5	12	104	632.62	632.00	0.10	98.75	26	0.16	1.82	3.42
55	6	12	87	638.58	638.00	0.09	99.69	15	0.21	1.59	3.01
55	7	12	1	640.00	640.00	0.00	100.00	1	0.25	0.36	0.65
55	8	12	1	640.00	640.00	0.00	100.00	0	0.31	0.00	0.35

[a] IBM 3090-600J computer used as fast serial machine.
[b] In seconds of CPU. Ttime exclusive of I/O times.

networks were obtained from a pre-established density δ defined as

δ = No. arcs in network/
no. arcs in corresponding complete network.

The densities used were 0.10 for the 100-vertex network and 0.05 for the 150-vertex network. Once the number of arcs was obtained for each network, the extreme points (vertices) of each arc and the corresponding length were randomly generated. The length of each arc is an integer value sampled from a uniform distribution defined in the range [40,60] for both networks. Floyd's algorithm was then used to obtain the corresponding distance matrices.

The population of each demand area was also randomly generated. These populations were sampled from a uniform distribution in the range [20,30] for the 100-vertex network and from a normal distribution with mean equal to 80 and standard deviation equal to 15 for the 150-vertex network.

An important decision that had to be made in relation to the data for the test problems was the values of p and S to be used for each problem. These parameters are not meaningful when taken separately, but their combined values determines the percentage of the total population that will be covered in each case. We decided to choose values of p and S that would generate a meaningful range of total population covered. The values chosen generated a range of percentage of total population covered varying from 70% to 100% for the 55-vertex and the 100-vertex networks and from 60% to 97% for the 150-vertex network.

Detailed computational results are shown in Tables 2, 3 and 4. For each problem, identified by the size of the network (n) and the values of p and S, we show the number of iterations of the subgradient optimization algorithm (Iter), the values of the upper and lower bounds available at the end of the Lagrangean heuristic (Ubound and Lbound, respectively) [In order to avoid showing negative values for bounds in Tables 2–4, the bounds shown in these tables are relative to the maximization form of MCLP (maximization of total population covered)], the gap between these bounds, expressed as a percentage of the upper bound,

$$\%\text{Dif} = \frac{\text{Ubound} - \text{Lbound}}{\text{Ubound}} \cdot 100\%,$$

the percentage of the total population covered, % Cov., the number of times the interchange heuristic was used to improve the value of v_{primal}, Ncalls, and the corresponding computing times. These computing times are broken down into time to compute the initial lower bound through the 'greedy'-interchange heuristic (T GI Hr.), to-

Table 3
Computational results – 100-vertex network [a]

n	p	S	Iter	Ubound	Lbound	% Dif	% Cov.	Ncalls	Comp. times [b]		
									T GI Hr.	T I Hr.	Ttime
100	8	50	474	1779.02	1703.00	4.27	69.43	60	0.73	14.86	51.69
100	10	50	495	1937.53	1870.00	3.49	76.23	71	0.99	22.99	62.90
100	12	50	471	2055.97	2002.00	2.63	81.61	66	1.24	26.45	64.26
100	8	65	464	2315.03	2143.00	7.43	87.36	54	0.71	16.66	53.81
100	10	65	200	2453.00	2314.00	5.67	94.33	8	0.98	3.52	20.40
100	12	65	200	2453.00	2433.00	0.82	99.18	8	1.31	4.75	22.03
100	8	80	372	2349.94	2170.00	7.66	88.46	48	0.69	13.63	43.56
100	10	80	200	2453.00	2360.00	3.79	96.21	8	1.01	3.25	20.54
100	12	80	51	2453.00	2453.00	0.00	100.00	3	1.41	1.72	7.41

[a] IBM 3090-600J used with vector and parallel capabilities.
[b] In seconds of CPU. Ttime exclusive of I/O times.

Table 4
Computational results – 150-vertex network [a]

n	p	S	Iter	Ubound	Lbound	% Dif	% Cov.	Ncalls	Comp. times [b]		
									T Gl Hr.	T I Hr.	Ttime
150	8	75	649	7382.10	7104.00	3.77	59.14	66	1.12	30.41	109.71
150	10	75	713	8701.55	8272.00	4.94	68.86	59	1.66	43.40	122.35
150	12	75	546	9848.78	9291.00	5.66	77.34	61	2.15	66.81	127.28
150	8	80	739	7711.06	7387.00	4.20	61.49	82	1.08	36.82	125.45
150	10	80	629	9032.53	8518.00	5.70	70.91	72	1.40	49.91	118.43
150	12	80	596	10094.04	9394.00	6.94	78.20	58	1.98	57.50	122.89
150	8	85	534	9231.96	8883.00	3.78	73.94	54	1.21	32.43	96.39
150	10	85	667	10266.68	9798.00	4.57	81.56	63	1.80	54.66	127.59
150	12	85	758	11094.62	10566.00	4.76	87.95	62	2.28	70.67	154.12
150	8	90	849	11091.80	10787.00	2.75	89.79	65	1.38	41.23	144.40
150	10	90	793	11661.84	11311.00	3.01	94.16	72	2.15	60.96	147.92
150	12	90	296	12013.00	11703.00	2.58	97.42	8	2.66	11.43	46.89

[a] IBM 3090-600J used with vector and parallel capabilities.
[b] In seconds of CPU. Ttime exclusive of I/O times.

tal time taken by the interchange heuristic in the Ncalls in which it was used to improve v_{primal} (T I Hr.) and total time of the heuristic (Ttime).

The results corresponding to the 55-vertex network are shown in Table 2. Except for three of the problems ($\{n = 55, p = 8, S = 8\}$, $\{n = 55, p = 8, S = 10\}$ and $\{n = 55, p = 3, S = 12\}$), a guaranteed optimal solution was always obtained for these problems. The computing times shown correspond to the 'super-computer' being used as a fast serial machine. We did run these problems using the vector and parallel capabilities of the 'super-computer', but the 'overhead' time required to use those capabilities outweighted the gains of faster computation; serial times therefore were the lowest for the 55-vertex problems. The number of subgradient iterations was generally low, although for a few problems it did reach the 500 limit.

Tables 3 and 4 show the results for the 100- and 150-vertex randomly generated networks. For these problems the use of the vector and parallel capabilities of the 'super-computer' produced a significant reduction in computing times; for some of the 150-vertex problems the times shown are approximately half of the corresponding serial times. It should be noted however that no special effort was made when writing the code to take advantage of vectorization and parallelism; we just used automatic vectorization and parallelism presently available in Fortran compilers.

Only for some of these larger problems it was possible to obtain and verify an optimal solution, but we were not able correlate the size of the gap to the parameters that define the problem (n, p and S). The number of subgradient iterations varied from a few to 849 iterations. It is possible to detect an increase in the number of iterations as problem size increases from a 55-vertex to an 150-vertex network.

It would be difficult to compare our computing times with those of Weaver and Church [12], given the time span between the two papers and the advances in computer technology in the last ten years. We are not claiming however that our method is faster than that of [12]; we only claim that our solutions are of better quality, especially for the larger problems (see next-to-last paragraph of Section 1).

5. Conclusion

In this paper we developed a Lagrangean heuristic for the maximal covering location prob-

lem and tested it in networks of up to 150 vertices. The procedure uses subgradient optimization to obtain lower bounds and vertex addition and substitution algorithms to obtain upper bounds for the problem. It was generally possible to verify optimality for the 55-vertex network problems, but for the larger problems a duality gap was usually present at the end heuristic. In the latter case the average duality gap was 3.97% for the 100-vertex network and 4.39% for the 150-vertex network. The maximum gap for all problems was 7.66%.

A 'super-computer' was used to test the heuristic and the maximum computing time was under 3 minutes of CPU in an IBM 3090-600J. For the 55-vertex network this computer was used as a fast serial machine, but for the larger problems the use of vectorization and parallelism cut computing times substantially, in spite of the fact that no special effort was made to take advantage of these capabilities when the coding was done.

The results we obtained indicate that Lagrangean relaxation is an important technique in the solution of location covering problems. Covering models studied in the literature make extensive use of linear programming relaxation as a solution method, taking advantage of the fact that the corresponding mathematical formulations are such that an integer solution is produced in the vast majority of the cases. The computational experience available in the literature with LP relaxation is however restricted to problems not larger than the 55-vertex network we used in this paper, and there is no guarantee that integer solutions will be obtained as frequently for the larger problems. In fact we conducted limited computational experiments with the LP relaxation of MCLP using some of the 100-vertex problems of Table 3, always obtaining fractional solutions for these problems.

Given the numerous covering models presently available in the literature, research on Lagrangean bounds for these problems appears to hold promise for further successful use. The use of Lagrangean relaxation-based branch-and-bound algorithms may also prove more effective than the LP-based searches that were developed for some of the covering problems.

References

[1] Christofides, N., and Eilon, S., "Algorithms for large scale travelling salesman problems", *Operational Research Quarterly* 23 (1972) 511–518.

[2] Church, R.L., and ReVelle, C.S., "The maximal covering location problem", *Papers of the Regional Science Association* 32 (1974) 101–118.

[3] Church, R.L., Current, J., and Storbeck, J., "A bicriterion maximal covering location formulation which considers the satisfaction of uncovered demand", *Decision Sciences* 22 (1991) 38–52.

[4] Eilon, S., and Galvão, R.D., "Single and double vertex substitution in heuristic procedures for the *p*-median problem", *Management Science* 24 (1978) 1763–1766.

[5] Garfinkel, R.S., Neebe, A.W., and Rao, M.R., "An algorithm for the *m*-median plant location problem", *Transportation Science* 8 (1974) 217–236.

[6] Kerningan, B.W., and Lin, S., "An efficient heuristic procedure for partitioning graphs", *Bell System Technical Journal* 49 (1970) 291–307.

[7] Lin, S., "Computer solutions of the travelling salesman problem", *Bell System Technical Journal* 44 (1965) 2245–2269.

[8] Pastor, J.T., and Alminana, M., "Una panorámica de las conexiones y aplicaciones del problema de localización con cubrimiento maximal" ("A panoramic view of the conexions and applications of the maximal covering location problem"), *Investigación Operativa* 3 (1993) 25–39.

[9] Pirkul, H., and Schilling, D., "The capacitated maximal covering location problem with backup service", *Annals of Operations Research* 18 (1989) 141–154.

[10] Swain, R., "A decomposition algorithm for a class of facility location algorithms", Ph.D. Thesis, Cornell University, Ithaca, NY, 1971.

[11] Teitz, M.B., and Bart, P., "Heuristic methods for estimating the generalized vertex median of a weighted graph", *Operations Research* 16 (1968) 955–961.

[12] Weaver, J.R., and Church, R.L., "A comparison of direct and indirect primal heuristic/dual bounding solution procedures for the maximal covering location problem", Unpublished paper, 1984.

[13] Whitaker, R.A., "A fast algorithm for the greedy interchange for large-scale clustering and median location problems", *INFOR* 21 (1983) 95–108.

[14] White, J., and Case, K., "On covering problems and the central facility location problem", *Geographical Analysis* 6 (1974) 281–293.