Extensions to a Lagrangean relaxation approach for the capacitated warehouse location problem

N. CHRISTOFIDES and J.E. BEASLEY
Department of Management Science, Imperial College, London, SW7 2BX, United Kingdom

Received July 1981
Revised February 1982

In this paper we present a lower bound for the capacitated warehouse location problem based upon the Lagrangean relaxation of a mixed-integer formulation of the problem, where we use subgradient optimisation in an attempt to maximise this lower bound. Problem reduction tests based upon this lower bound and the original problem are given. Incorporating this bound and the reduction tests into a tree search procedure enables us to solve problems involving up to 50 warehouses and 150 customers.

1. Introduction

The capacitated warehouse location problem is the problem of locating a number of warehouses which have to service a set of customers, at minimum cost, where each customer has an associated demand and there are constraints on the total demand that can be met from a warehouse.

This problem has been considered by a number of authors in the literature e.g. Davis and Ray [4], Sa [14], Ellwein and Gray [5], and Akinc and Khumawala [1]. The approach of these authors is based on the linear programming relaxation of a mixed-integer formulation of the problem which is used as a bound for a tree search procedure, together with reduction tests based on criteria for opening/closing warehouses.

In recent years a number of papers applying Lagrangean relaxation to the problem have appeared. Geoffrion and McBride [8] applied Lagrangean relaxation to the capacitated problem with arbitrary additional constraints. The main thrust of their work was towards understanding Lagrangean relaxation and only a small amount of computational experience on problems drawn from the literature was reported. Nauss [11] presented a similar procedure for the capacitated warehouse location problem as we present in this paper. His results indicate that his procedure is currently the most effective one developed for the problem. The main differences between our work and his are

(i) a constraint limiting the number of open warehouses (as used by Neebe [12]),

(ii) use of a stronger set of reduction tests,

(iii) a constraint ensuring that the demand allocated to an open warehouse exceeds a certain level,

(iv) computational results for much larger problems than have been reported by other authors.

Guignard and Spielberg [9] have developed a direct dual method for the problem – their procedure being motivated by the success of a similar approach for the uncapacitated problem due to Bilde and Krarup [2] and Erlenkotter [6]. They mainly present results indicating the quality of the bound they derive. Van Roy [15,16] has developed a bound based upon cross decomposition which dominates the bound of Guignard and Spielberg.

2. Problem formulation

We formulate the capacitated warehouse location problem as a mixed-integer program. Let

q_j be the demand of customer j,

d_{ij} be the cost of supplying all of the demand of j from a warehouse located at i with $d_{ij} \geq 0$,

X_{ij} be the fraction of the demand of j (q_j) that is supplied from a warehouse at i so that $0 \leq X_{ij} \leq 1$,

Q_i be the capacity of warehouse i (i.e. the limit

North-Holland Publishing Company
European Journal of Operational Research 12 (1983) 19–28

on the total demand that can be allocated to a warehouse located at i),

L_i be the lower limit on the demand that must be allocated to warehouse i (if it is open),

P_L be a lower limit on the number of open warehouses,

P_U be an upper limit on the number of open warehouses,

F_i be the fixed cost associated with warehouse i.

Define

$$Y_i = \begin{cases} 1 & \text{if warehouse } i \text{ is open,} \\ 0 & \text{otherwise,} \end{cases}$$

then with n customer/potential warehouse positions the program is

$$\min z = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} X_{ij} + \sum_{i=1}^{n} F_i Y_i, \tag{1}$$

s.t.

$$\sum_{i=1}^{n} X_{ij} \geq 1, \qquad j = 1,\ldots,n, \tag{2}$$

$$L_i Y_i \leq \sum_{j=1}^{n} q_j X_{ij} \leq Q_i Y_i, \quad i = 1,\ldots,n, \tag{3}$$

$$P_L \leq \sum_{i=1}^{n} Y_i \leq P_U, \tag{4}$$

$$0 \leq X_{ij} \leq 1, \qquad i = 1,\ldots,n, \quad j = 1,\ldots,n, \tag{5}$$

$$Y_i \in (0,1), \qquad i = 1,\ldots,n. \tag{6}$$

Equation (2) ensures that the demand of every customer is satisfied. Equation (3) ensures that no customer can be supplied from a closed warehouse and that the total demand supplied from a warehouse lies within the limits imposed upon it. Equation (4) ensures that the number of open warehouses lies between P_L and P_U (note that this equation is useful because both P_L and P_U can be updated during our solution procedure to provide tighter limits on the number of open warehouses). Equation (5) provides an upper bound on the allocation variables (X_{ij}) whilst equation (6) is the integrality constraint. We will assume that there exists a feasible solution (Y_i), (X_{ij}) to the above program.

Note that in this formulation we treat all customer positions as potential warehouse sites for simplicity – if at any customer position j a warehouse cannot be located, set F_j to infinity to preclude the possibility of opening a warehouse there in the optimal solution. This problem can also be regarded as a fixed charge transportation (network flow) problem in which the customer demands represent sinks and the capacity for each warehouse represents a source – there being a lower limit on the amount supplied from each source and also a constraint on the number of sources.

If the allocation variables (X_{ij}) are restricted to be zero-one variables, so that a customer must be supplied from one and only one warehouse, then the problem becomes much harder to solve. At present if a set of open warehouses are given then the optimal allocation of customers to warehouses can be derived by solving a network flow problem. However with zero-one allocation variables the resolution of the allocation of customers to warehouses, given the set of open warehouses, itself involves some tree search procedure (and is a special case of generalised assignment problem (Ross and Soland [13])).

3. Lower bounds on the problem derived from Lagrangean relaxation

To solve the capacitated warehouse location problem we develop a lower bound, for use in a tree search procedure, from a Lagrangean relaxation of the problem.

We introduce Lagrange multipliers λ_j ($\geq 0, j = 1,\ldots,n$) for the allocation constraint (eq. (2)) and obtain the Lagrangean dual program

$$\min z_D = \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ij} - \lambda_j) X_{ij} + \sum_{i=1}^{n} F_i Y_i + \sum_{j=1}^{n} \lambda_j, \tag{7}$$

s.t.

$$L_i Y_i \leq \sum_{j=1}^{n} q_j X_{ij} \leq Q_i Y_i, \quad i = 1,\ldots,n, \tag{8}$$

$$P_L \leq \sum_{i=1}^{n} Y_i \leq P_U, \tag{9}$$

$$0 \leq X_{ij} \leq 1, \qquad i = 1,\ldots,n, \quad j = 1,\ldots,n, \tag{10}$$

$$Y_i \in (0,1), \qquad i = 1,\ldots,n. \tag{11}$$

3.1. Solution of the Lagrangean dual program

The Lagrangean dual program (eq. (7) to (11)) can be solved easily.

Consider the effect of setting Y_i to one (i.e. opening a warehouse at i), then the best set of allocations of customers j to this warehouse is given by

$$\min \sum_{j=1}^{n} (d_{ij} - \lambda_j) X_{ij}, \tag{12}$$

s.t.

$$L_i \leqslant \sum_{j=1}^{n} q_j X_{ij} \leqslant Q_i, \tag{13}$$

$$0 \leqslant X_{ij} \leqslant 1, \quad j = 1,\ldots,n. \tag{14}$$

This program is the linear programming relaxation of a constrained knapsack problem and can be easily solved without recourse to the simplex method by the procedure outlined below.

(a) Order the X_{ij} by increasing $((d_{ij} - \lambda_j)/q_j)$, i.e. by increasing order of the average cost per unit supplied from i to j.

(b) Proceed along this list supplying customers from warehouse i (setting the X_{ij} values accordingly) until we have supplied L_i units.

(c) Continue along this list setting each X_{ij} in turn to one (i.e. supplying j completely from i) provided that there is enough capacity left at i to supply q_j taking into account the other customers already supplied from i and $(d_{ij} - \lambda_j) \leqslant 0$. When we encounter a customer j for whom there is not enough spare capacity at i to supply j completely set X_{ij} as high as possible to exhaust the capacity at i.

(d) Let the set of values for X_{ij} chosen as in (b) and (c) be X_{ij}^* (all X_{ij} not given an explicit value in (b) and (c) being zero). Let α_i represent the contribution to the Lagrangean dual objective function resulting from setting Y_i to one, then

$$\alpha_i = F_i + \sum_{j=1}^{n} (d_{ij} - \lambda_j) X_{ij}^*. \tag{15}$$

Computing α_i for all i as above the Lagrangean dual program reduces to

$$\min z_D = \sum_{i=1}^{n} \alpha_i Y_i + \sum_{j=1}^{n} \lambda_j, \tag{16}$$

s.t.

$$P_L \leqslant \sum_{i=1}^{n} Y_i \leqslant P_U, \tag{17}$$

$$Y_i \in (0,1), \quad i = 1,\ldots,n. \tag{18}$$

This zero-one program can be solved by inspection as follows:

(i) Form a list of the α_i arranged in ascending order.

(ii) Set Y_i to one for the first P_L warehouses in this list.

(iii) If $P_L = P_U$ go to (v).

(iv) Proceed along the list from the $(P_L + 1)$th warehouse setting Y_i to one until either (a) have opened P_U warehouses or (b) the α_i values become $\geqslant 0$.

(v) Let the set of values for the Y_i chosen as above be Y_i^* (all i not picked above having $Y_i^* = 0$).

Then the optimal solution to the Lagrangean dual program is (Y_i^*, $i = 1,\ldots,n$ and X_{ij}^*, $i = 1,\ldots,n$, $j = 1,\ldots,n$) with the solution value z_D^* being given by

$$z_D^* = \sum_{i=1}^{n} \alpha_i Y_i^* + \sum_{j=1}^{n} \lambda_j. \tag{19}$$

The set of open warehouses M^* is given by

$$M^* = (i \mid Y_i^* = 1). \tag{20}$$

We see from the solution to the Lagrangean program how tightening P_L and P_U will potentially improve the lower bound given by the Lagrangean relaxation for any set of Lagrange multipliers.

4. Determination of a feasible solution

A feasible solution to the original problem is not automatically available from the solution to the Lagrangean problem given above. The advantage in searching for a feasible solution is, of course, that if an improved feasible solution is found the size of the tree search is reduced. We adopted the procedure given below to search for a feasible solution.

(a) Form a list of the customers – the first part of this list consisting of the customers in M^* arranged in descending demand order, and the second part of the list being the remaining customers, also arranged in descending demand

order. (Note that for all the problems we considered each warehouse also corresponded to a customer.)

(b) Proceed along this list picking each customer j in turn and performing (c) below.

(c) Supply as much as possible of the demand of j from the cheapest warehouse i for which $X_{ij}^* \neq 0$ having regard to the other customers already supplied from i. If j is not completely supplied then attempt to supply it from the second cheapest warehouse for which $X_{ij}^* \neq 0$, then the third cheapest etc. Eventually if all warehouses for which $X_{ij}^* \neq 0$ are exhausted attempt to supply j from the remaining open warehouses for which $X_{ij}^* = 0$, these warehouses being considered in ascending order of their cost of supplying j. If after considering all possible open warehouses in M^* we have failed to completely supply j then the procedure cannot find a feasible solution, so quit.

(d) After all the customers have been supplied we may have a solution which is feasible (i.e. satisfies eqs. (2) to (6)). If so then we can update z_{UB} the upper bound on the problem (corresponding to a feasible solution) accordingly.

The purpose of making such a procedure dependent upon the X_{ij}^* values, rather than just M^* alone, is that we found that often M^* did not change between successive Lagrangean dual solutions and so effort would be expended recalculating the same feasible solution if the procedure depended only upon M^*.

5. Problem reduction

Reduction can be achieved through the use of an upper bound z_{UB} to the optimal solution and penalties based upon the Lagrangean dual solution as shown below.

Define

$$K = [i \mid F_i \neq \infty], \tag{21}$$

$$K_1 = [i \mid Y_i = 1], \tag{22}$$

$$K_2 = [i \mid Y_i = 0], \tag{23}$$

i.e. K the set of potential warehouse positions and K_1/K_2 the set of warehouses that have been opened/closed at the current stage in the procedure. Note that this implies that we automatically include K_1 in M^* (and exclude K_2 from M^*) in the solution of the Lagrangean dual program.

Then the penalties we can calculate to reduce the size of the problem are shown below. Note that if the penalty value forces the lower bound above z_{UB} then the condition that results in that penalty value can never occur in the optimal completion of the current stage in the procedure.

5.1. Open penalties

The penalty for bringing a warehouse $i \notin M^*$ into the dual solution (i.e. opening i) is given by

$$\alpha_i \quad \text{if } |M^*| \neq P_L \text{ and } |M^*| \neq P_U, \tag{24}$$

$$\alpha_i - \max_{j \in M^* - K_1} (\alpha_j) \quad \text{if } |M^*| = P_U, \tag{25}$$

$$\alpha_i - \max\Big(0, \max_{j \in M^* - K_1} (\alpha_j)\Big)$$
$$\text{if } |M^*| \neq P_U \text{ and } |M^*| = P_L. \tag{26}$$

These penalties preserve the condition that there are between P_L and P_U open warehouses in the optimal solution.

5.2. Close penalties

The penalty incurred in closing a warehouse $i \in M^* - K_1$ is given by

$$-\alpha_i + \min\Big(0, \min_{j \in K - M^* - K_2} (\alpha_j)\Big) \quad \text{if } |M^*| \neq P_L, \tag{27}$$

$$-\alpha_i + \min_{j \in K - M^* - K_2} (\alpha_j) \quad \text{if } |M^*| = P_L. \tag{28}$$

5.3. Penalty on the number of open warehouses

We can see that closing the gap between P_L and P_U strengthens the lower bound and also the open/close penalties. Obviously the dual solution value obtained if we enforce the condition that there are exactly p open warehouses in the optimal solution is given by

$$\sum_{j=1}^{n} \lambda_j + \sum_{i \in K_1} \alpha_i + \text{the } p - |K_1| \text{ smallest } \alpha_j \quad (j \in K - K_1 - K_2). \tag{29}$$

It is clear that if the value of (29) exceeds the upper bound z_{UB} when $p = P_L$ ($p = P_U$) we can

increase P_L (decrease P_U) by one without affecting the optimal solution to the original problem.

5.4. Penalty on warehouse capacity limits

It is clear from the solution for the best set of allocations of customers to a particular (open) warehouse given in Section 3.1 that we can find a penalty associated with a warehouse i being open with exactly D units of demand being supplied from it. By investigating all values of D where $L_i \leqslant D \leqslant Q_i$ we can update L_i and Q_i.

In our computational tests we investigated this penalty but found that the benefit gained from it in terms of earlier convergence to an optimal solution did not outweigh the effort spent in calculation and so in the computational results reported in this paper these penalties were not used.

5.5. Other reduction tests

We also investigated two further reduction tests developed by other authors that can be used for fixing warehouses open. Essentially these tests say that if, for a particular warehouse, the saving in variable allocation cost from opening that warehouse outweighs the fixed cost incurred in opening it then it can be fixed open.

Note however that reduction tests of this type *cannot* be used if the optimal solution to the original program (eqs. (1) to (6)) is changed by the removal of the constraint that limits the number of open warehouses (eq. (4)) or by the removal of the lower limits (L_i) on the demands that must be supplied from the warehouses.

The two reduction tests we investigated were

(a) Let $Z[K]$ represent the optimal value (excluding fixed costs) of the capacitated warehouse location problem with the set K of warehouses open and all other warehouses closed – this can be obtained very easily as the problem (eqs. 1 to 6) reduces to a simple network flow problem when the warehouses to be opened are known.

Then $Y_i = 1$ $(i \in K - K_1 - K_2)$ if

$$Z[K - K_2 - [i]] - Z[K - K_2] \geqslant F_i. \tag{30}$$

For an explanation of this test see Akinc and Khumawala [1]

(b) A further reduction test that is less expensive computationally than the one given above is given by

$$\text{define } c_{ij} = \min_{k \in K - K_2 - [i]} (d_{kj}), \tag{31}$$

i.e. c_{ij} the minimum cost of allocation of customer j other than to warehouse i. Then if the optimal value of the program given below exceeds F_i warehouse i should be fixed open. The program is

$$\max \sum_{j=1}^{n} [c_{ij} - d_{ij}] X_{ij}, \tag{32}$$

s.t.

$$\sum_{j=1}^{n} q_j X_{ij} \leqslant Q_i, \tag{33}$$

$$0 \leqslant X_{ij} \leqslant 1, \quad j = 1, \ldots, n, \tag{34}$$

Again see Akinc and Khumawala [1] for an explanation of this test. Note that the above program is easily solved in a similar manner to that described in Section 3.1.

We found that the computational effort involved in performing reduction test (a) – which entailed solving a number of network flow problems, was not justified in terms of increased convergence to an optimal solution. This was especially true if we performed reduction test (b) in place of reduction test (a). Accordingly in all the computational results reported in this paper reduction test (a) was not used.

6. The subgradient procedure

The subgradient procedure incorporating the penalty tests given in the previous section is as follows:

(1) Choose an initial value for the λ_j (arbitrary) – we used

$$\lambda_j = \min_{i \in K - K_2} (d_{ij}), \quad j = 1, \ldots, n. \tag{35}$$

(2) Determine an initial value for z_{UB} – the upper bound on the problem. This can be done using any heuristic for the problem.

(3) For the current set of Lagrange multipliers (λ_j) determine the optimal Lagrangean dual solution z_D^* (eq. 19) with M^* the set of open warehouses (eq. 20) and X_{ij}^* the associated variable values. Update Z_D the best lower bound found accordingly.

(4) Use the heuristic of Section 4 to derive a feasible solution from M^* and (X_{ij}^*) – update z_{UB} accordingly.

(5) Stop if $z_{UB} = Z_D$, i.e. the best lower bound and the upper bound (corresponding to a feasible solution) coincide else go to (6).

(6) Perform the penalty reduction tests of Sections 5.1, 5.2 and 5.3 for opening/closing warehouses and tightening the gap between P_L and P_U. If we close a warehouse as a result of these tests then perform reduction test (b) of Section 5.5.

(7) Define the subgradient vector S by

$$S_j = 1 - \sum_{i=1}^{n} X_{ij}^*, \quad j = 1,\ldots,n. \tag{36}$$

Stop if $S_j = 0, j = 1,\ldots,n$, else go to (8).

(8) Calculate the step size t for use in updating the Lagrange multipliers by

$$t = \pi(z_{UB} - z_D^*)/\|S\|^2 \tag{37}$$

where π a constant $(0 < \pi \leq 2)$ and $\|S\|$ any norm of the subgradient vector – we used the Euclidean norm $(\sum_{j=1}^{n} S_j^2)^{1/2}$.

(9) Update the multipliers by

$$\lambda_j = \max\Big(\min_{i \in K - K_2} (d_{ij}), \lambda_j + tS_j \Big), \quad j = 1,\ldots,n. \tag{38}$$

(10) Go to (3) unless sufficient subgradient iterations have been done in which case stop.

In choosing a value for π we followed the approach of Held, Wolfe, and Crowder [10] in letting $\pi = 2$ for $2n$ iterations and then successively halving both the value of π and the number of iterations until the number of iterations reached a threshold value of 5, π was then halved every 5 iterations.

7. The tree search procedure

In the event that the subgradient procedure does not optimally solve the problem a reduction in problem size will probably have been achieved because the penalty tests will have fixed certain warehouses open or closed. Any of the existing tree search procedures for the capacitated warehouse location problem could then be used on the reduced problem to obtain an optimal solution (albeit with certain modifications for the additional constraints in our formulation). We used a depth-first tree search procedure based on the Lagrangean relaxation of the problem discussed above. The branching strategy used was to pick the warehouse j corresponding to

$$\alpha_j = \min_{i \in K - K_1 - K_2} (\alpha_i) \tag{39}$$

at any node of the tree (ties arbitrarily broken) and to branch by opening warehouse j (as intuitively j corresponds to the warehouse most likely to be open in the optimal completion of the current tree node).

We indicate below how the tree search was structured to reflect the trade-off between possible improvement to the bound by continuing the subgradient procedure at a tree node, and branching.

7.1. The initial node

At the initial tree node we first carried out the reduction test (b) of Section 5.5 in an attempt to fix open some warehouses. (In this discussion we will assume that the reduction test (b) given in Section 5.5 is applicable.) The subgradient procedure was then carried out until either of (i), (ii) or (iii) below were satisfied

(i) all subgradients were zero,

(ii) the upper bound (z_{UB}) and the highest lower bound (Z_D) attained coincided,

(iii) π fell below 0.000001.

If the subgradient procedure failed to converge to an optimal feasible solution then the set of Lagrange multipliers associated with the highest lower bound (Z_D) found were recalled. The optimal allocation of customers to warehouses was found, for the open warehouse set associated with these multipliers, by solving a network flow problem. The solution value obtained was used to update z_{UB}, the upper bound on the problem.

A further thirty iterations of the subgradient procedure with $\pi = 0.1$ were then performed in an attempt to take advantage of any improved feasible solution obtained.

7.2. The tree search nodes

Each tree node had associated with it a vector which contained information on the state of the warehouses (which ones were fixed open/closed

and which ones were still undecided), the set of Lagrange multipliers associated with the best lower bound found at that node and the values for P_L and P_U at that node.

The subgradient procedure was carried out at each tree node for 30 iterations with $\pi = 0.1$ using a target value in eq. (37) 1% above the upper bound – these parameters being decided upon after exploring a number of possibilities.

The initial set of multipliers used at any node was the best set associated with the tree node from which branching was taking place and the reduction tests of Sections 5.1, 5.2 and 5.3 were carried out at each subgradient step with the reduction test (b) of Section 5.5 being performed at the start of each tree node and thereafter whenever a warehouse was fixed closed.

7.3. Evaluation of terminal nodes

A terminal node in the tree search is one where there are no undecided warehouses – i.e. all warehouses have been fixed open or closed. As mentioned previously, once the warehouse sites have been fixed then the problem reduces to a network flow problem which can be easily solved to give the optimal allocation of customers to warehouses.

8. Results

The algorithm was programmed in FORTRAN and run on a CDC 7600 machine (using the FTN compiler) on a number of test problems drawn from the literature and some randomly generated problems. Table 1 presents a summary of these

Table 1
Test problem details

Problem set	Number of potential warehouse sites	Number of customers	Warehouse capacity	Fixed cost per warehouse ($'000)
IV	16	50	5000	7.5/12.5/17.5/25.0
V	16	50	10000	17.5
VI	16	50	15000	7.5/12.5/17.5/25.0
VII	16	50	58268	7.5/12.5/17.5/25.0
VIII	25	50	5000	7.5/12.5/17.5/25.0
IX	25	50	15000	7.5/12.5/17.5/25.0
X	25	50	58268	7.5/12.5/17.5/25.0
XI	50	50	5000	7.5/12.5/17.5/25.0
XII	50	50	15000	7.5/12.5/17.5/25.0
XIII	50	50	58268	7.5/12.5/17.5/25.0
XIV	25	75	25/50/100	Random
XV	35	100	25/50/100	Random
XVI	50	150	25/50/100	Random

Table 2
Results for problems not requiring branching

Problem number	Number of subgradient iterations	Time CDC 7600 seconds
VI -1	9	0.5
-2	13	0.5
-4	108	1.3
VII -1	9	0.4
-2	13	0.4
-3	99	1.5
-4	14	0.4
IX -1	62	1.3
X -1	17	0.6
-2	28	0.6
-3	176	1.8
-4	23	0.4
XII -1	123	3.1
XIII -1	119	3.7
-2	112	3.0
-3	156	3.1
-4	96	2.1
XIV -2	356	6.2
-3	11	1.2
XV -2	466	16.3
-3	201	5.8
XVI -2	641	55.6

Table 3
Results for problems requiring branching

Problem number	Initial tree node							Number of tree nodes	Number of terminal nodes	Total time CDC 7600 seconds
	Final P_I	Final P_U	Number fixed open	Number undecided	Duality gap (%)	Number of iterations	Time CDC 7600 seconds			
IV -1	12	16	10	6	0.927	295	3.7	52	26	8.7
-2	12	16	9	7	0.709	295	3.7	34	15	6.9
-3	12	14	8	8	0.852	295	3.6	32	14	6.7
-4	12	13	8	8	0.630	295	3.6	12	6	4.9
V -1	8	9	7	3	0.044	295	2.7	4	3	3.1
VI -3	5	8	3	7	0.132	295	2.1	12	4	2.8
VIII -1	12	21	11	12	0.177	295	4.7	24	7	7.3
-2	12	17	10	8	0.125	295	4.6	16	6	6.3
-3	12	16	11	6	0.092	295	4.7	8	2	5.8
-4	12	17	5	18	0.188	295	4.7	58	25	12.0
IX -2	8	12	8	5	0.075	295	2.9	8	2	4.1
-3	5	11	4	12	0.199	295	3.0	30	11	6.6
-4	4	10	3	11	0.404	295	2.9	54	16	8.7
XI -1	16	18	16	2	0.036	295	6.0	4	2	7.1
-2	12	17	9	10	0.131	295	6.1	38	14	12.6
-3	12	19	7	14	0.310	295	6.6	144	55	30.2
-4	12	18	2	48	0.310	295	6.6	214	87	47.3
XII -2	7	13	5	10	0.076	295	3.9	16	3	5.7
-3	5	12	3	11	0.110	295	3.8	36	13	8.0
-4	4	11	2	24	0.422	295	3.8	152	42	20.7
XIV -1	15	15	14	2	0.003	385	9.1	2	2	9.9
IX -1	20	20	18	3	0.005	495	25.1	4	3	28.0
XVI -1	31	31	30	2	0.001	670	97.2	2	2	109.0
-3	8	8	7	2	0.004	670	29.7	2	2	33.8

problems, the notation for which follows that of Akinc and Khumawala [1] where further details of the problems can be found. We extended the problems they considered by defining problem sets XI, XII and XIII which are the same as problem sets VIII, IX and X respectively except that for these new problem sets all customer locations were potential warehouse locations.

Problem sets XIV, XV and XVI were produced by randomly generating an inter-location distance matrix (which was subjected to Floyd's [7] algorithm to ensure that the triangularity condition was satisfied) and taking the cost per unit of demand supplied as directly proportional to the distance travelled. For each customer the demand was a randomly generated integer in the range [1,10], the capacity for each warehouse was as shown in Table 1 and the fixed cost for each warehouse i was defined by taking a random number r_i in the range [0.75, 1.00] and setting $F_i \propto r_i Q_i$.

For all the problems considered no lower limits on the demand that must be allocated to a warehouse were enforced and an initial value P_L was obtained by calculating the minimum number of warehouses needed to supply all the customers with P_U being set initially to the total number of warehouses available.

An initial upper bound for each problem was produced by generating five random warehouse sets, each containing the minimum number of warehouses necessary to supply all the customers, and evaluating the cost of these sets by solving the associated network flow problem. The lowest of these costs was used as the initial upper bound on the problem.

Table 2 gives the results for those problems which terminated at the initial tree node with an optimal feasible solution and hence did not require branching. The time quoted is in CDC 7600 seconds and includes setup and input/output time as well as the time needed to generate the initial upper bound for the problem. We use IV-1 to refer to the first problem in problem set IV (i.e. with a fixed cost of 7500 for all warehouses), IV-2 to refer to the second problem (fixed cost 12500) etc.

As might be expected from the success of a similar method for the p-median problem (Christofides and Beasley [3]) the problem sets VII, X and XIII that were effectively uncapacitated were all solved without any branching being necessary.

Table 3 gives the results for the remaining problems that did require branching. The duality gap at the initial tree node is measured by

(optimal value − maximum lower bound at initial node)/(optimal value) × 100%.

It will be seen from that table that all the duality gaps were under 1% for problems that required branching.

Comparing our results with those of Nauss [11] on the problems we both attempted (problem sets IV, V, VI, VIII and IX) is difficult due to the way he has presented his results. However we note that for these common problems he reported an average CPU time of 13.0 seconds on an IBM 370/168 (excluding input/output time) with 12 network flow problems being solved per optimal solution obtained, whereas we took 5.1 seconds on a CDC 7600 (including input/output time) with 8.1 network flow problems solved per optimal solution obtained. It is perhaps worth noting here that on average 27% of the time needed to solve a problem in our results was taken up with the solution of network flow problems.

Comparing our results with those of Van Roy [15] (who merely presents details of the bound his cross decomposition algorithm derives) it is clear that his approach is more effective in the cases where convergence to an optimal solution is obtained (and hence no branching is necessary). However for the eight problems we both considered where such convergence was not obtained (problems VI-3, VIII-1/2/3/4, and IX-2/3/4) he reports an average duality gap of 0.2% (total time 6.3 seconds on an IBM 3033) whilst we obtained a lower average duality gap of 0.17% (total time 29.6 seconds on a CDC 7600).

References

[1] U. Akinc and B.M. Khumawala, An efficient branch and bound algorithm for the capacitated warehouse location problem, *Management Sci. 23* (1977) 585–594.

[2] O. Bilde and J. Krarup, Sharp lower bounds and efficient algorithms for the simple plant location problem, *Annals Discrete Math. 1* (1977) 79–97.

[3] N. Christofides and J.E. Beasley, A tree search algorithm for the p-median problem, *European J. Operational Res. 10* (1982) 196–204.

[4] P.S. Davis and T.L. Ray, A branch-bound algorithm for the capacitated facilities location problem, *Naval Res. Logist. Quart. 16* (1969) 331–343.

[5] L.B. Ellwein and P. Gray, Solving fixed charge location-allocation problems with capacity and configuration constraints, *AIIE Trans. 3* (1971) 290–298.

[6] D. Erlenkotter, A dual-based procedure for uncapacitated facility location, *Operations Res., 26* (1978) 992–1009.

[7] R.W. Floyd, Algorithm 97 – shortest path, *Comm. ACM 5* (1962) 345.

[8] A.M. Geoffrion and R. McBride, Lagrangean relaxation applied to capacitated facility location problems, *AIIE Trans. 10* (1978) 40–47.

[9] M. Guignard and K. Spielberg, A direct dual method for the mixed plant location problem with some side constraints, *Math. Programming 17* (1979) 198–228.

[10] M. Held, P. Wolfe and H.P. Crowder, Validation of subgradient optimisation, *Math. Programming 6* (1974) 62–88.

[11] R.M. Nauss, An improved algorithm for the capacitated facility location problem, *J. Operational Res. Soc. 29* (1978) 1195–1202.

[12] A.W. Neebe; A branch and bound algorithm for the *p*-median transportation problem, *J. Operational Res. Soc. 29* (1978) 989–995.

[13] G.T. Ross and R.M. Soland, Modelling facility location problems as generalised assignment problems, *Management Sci. 24* (1977) 345–357.

[14] G. Sa, Branch and bound and approximate solutions to the capacitated plant location problem, *Operations Res. 17* (1969) 1005–1016.

[15] T.J. Van Roy, A cross decomposition algorithm for capacitated facility location, *Operations Res.*, to appear.

[16] T.J. Van Roy, Cross decomposition for integer programming, *Math. Programming 25* (1983) 46–63.