Journal of the Operational Research Society (2003) 54, 11–20 0160-5682/03 $15.00
www.palgrave-journals.com/jors

A large-scale application of the partial coverage uncapacitated facility location problem

FJ Vasko[1,2]*, DD Newhart[2], KL Stott Jr[2] and FE Wolf[2]
[1]*Mathematics & CIS Department, Kutztown University, USA; and* [2]*Homer Research Laboratories, Bethlehem Steel Corporation, Bethlehem, USA*

The traditional, uncapacitated facility location problem (UFLP) seeks to determine a set of warehouses to open such that all retail stores are serviced by a warehouse and the sum of the fixed costs of opening and operating the warehouses and the variable costs of supplying the retail stores from the opened warehouses is minimized. In this paper, we discuss the partial coverage uncapacitated facility location problem (PCUFLP) as a generalization of the uncapacitated facility location problem in which not all the retail stores must be satisfied by a warehouse. Erlenkotter's dual-ascent algorithm, DUALOC, will be used to solve optimally large (1600 stores and 13 000 candidate warehouses) real-world implemented PCUFLP applications in less than two minutes on a 500 MHz PC. Furthermore, a simple analysis of the problem input data will indicate why and when efficient solutions to large PCUFLPs can be expected.
Journal of the Operational Research Society (2003) **54,** 11–20. doi:10.1057/palgrave.jors.2601469

Keywords: uncapacitated facility location; partial coverage; real-world application

Introduction

The customary definition of the uncapacitated facility location problem (UFLP) involves the determination of which candidate warehouses to open in order to service all demand locations (typically referred to as retail stores) such that the sum of the fixed costs of opening and operating the warehouses and the variable costs of supplying the retail stores from the opened warehouses are minimized. This problem has been widely studied over the last three decades. Daskin[1] as well as ReVelle and Laporte[2] discuss several new models and research prospectives for the plant location problem. More recently Gourdin *et al*[3] discuss a generalization of the UFLP that involves client matching.

In this paper, we discuss the partial coverage uncapacitated facility location problem (PCUFLP) as another generalization of the UFLP in which not all the retail stores must be satisfied by a warehouse. The PCUFLP is analogous to the ROI plant location problem[2] in that neither problem formulation requires that all retail stores be serviced by a warehouse. Also, the PCUFLP corresponds to the partial covering P-center problem and the partial set covering problems defined by Daskin and Owen.[4] After providing the mathematical formulation of the PCUFLP, an inventory sizing application of the PCUFLP will be discussed. Next, large real-world implemented applications of the PCUFLP will be solved using Erlenkotter's[5] dual-ascent algorithm, DUALOC, for the UFLP. The highly efficient solution times obtained by this classic algorithm (c. 1976) for these applications will be explained by analysing the PCUFLP input generated by the applications. Further testing of DUALOC will demonstrate how it can be applied heuristically to solve large PCUFLP regardless of the input data structure.

**Correspondence: FJ Vasko, Homer Research Laboratories, Bethlehem Steel Corporation, Bethlehem, PA 18016, USA.*
E-mail: vasco@kutztown.edu

The PCUFLP formulation

The mathematical formulation for the uncapacitated facility location problem is given below.

UFLP formulation

$$\text{Minimize } \sum\sum c_{ij}x_{ij} + \sum f_j y_j \tag{1}$$

subject to

$$\sum a_{ij}x_{ij} = 1, \qquad i = 1, \ldots, m \tag{2}$$

$$y_j - x_{ij} \geqslant 0, \qquad i = 1, \ldots, m \quad j = 1, \ldots, n \tag{3}$$

$$x_{ij} \geqslant 0, \qquad i = 1, \ldots, m \quad j = 1, \ldots, n \tag{4}$$

$$y_j \in \{0,\ 1\}, \qquad j = 1, \ldots, n. \tag{5}$$

In this formulation, a_{ij} is 1 if retail store i can be supplied from warehouse j and 0 if it cannot; c_{ij} is the variable cost of supplying store i from warehouse j. The number of retail stores is m and the number of candidate warehouse locations is n. The f_j values are the fixed costs of opening and operating the warehouses. The value of x_{ij} is 1 if store i is to be supplied from warehouse j and 0 otherwise. The value of y_j is 1 if warehouse j is in the solution and 0 otherwise. It is well known that if the y_j values are restricted to either 0 or 1, then the x_{ij} values will not take on fractional values. Therefore, the x_{ij} values do not have to be explicitly restricted to only 0 or 1.

The PCUFLP is a generalization of the UFLP in which not all the retail stores need to be serviced by a warehouse. Specifically, we introduce a 'universal' warehouse that can 'service' all retail stores and has a fixed cost of zero. The variable cost of servicing a retail store from the universal warehouse is the lost sales from that store. The universal warehouse is simply a 'dummy' facility that allows us to not service (shut down or never open) retail stores that are not economically feasible. The mathematical formulation for the PCUFLP problem is given below.

PCUFLP formulation

$$\text{Minimize} \sum\sum c_{ij}x_{ij} + \sum f_j y_j + \sum p_i U_i \tag{6}$$

subject to

$$\sum a_{ij}x_{ij} + U_i = 1, \qquad i = 1, \ldots, m \tag{7}$$

$$y_j - x_{ij} \geqslant 0, \qquad i = 1, \ldots, m \quad j = 1, \ldots, n \tag{8}$$

$$x_{ij} \geqslant 0, \qquad i = 1, \ldots, m \quad j = 1, \ldots, n \tag{9}$$

$$0 \leqslant U_i \leqslant 1, \qquad i = 1, \ldots, m \tag{10}$$

$$y_j \in \{0,\ 1\}, \qquad j = 1, \ldots, n. \tag{11}$$

All variables that also occur in the UFLP have the same definitions as given previously. Specific to the PCUFLP formulation, the p_i parameters represent the lost sales of retail store i, if it is 'serviced' by the universal warehouse. Constraint set (7) ensures that all retail stores can be serviced by the universal warehouse. It is easy to show that the U_i variables that must be between 0 and 1 (constraint set (10)) will not take on fractional values. In other words, U_i is 1 if retail store i is 'serviced' by the universal warehouse and zero otherwise. Thus the PCUFLP is a generalization of the UFLP; because if all the $U_i = 0$, then the PCUFLP reduces to a UFLP.

PCUFLP formulation for inventory sizing applications

Sheet steel for such diverse applications as automobile body panels, appliance panels, and steel storage sheds are sold to customers as master steel coils. The customer, in turn, slits or shears (or both) the master coils into his specific requirements. The master steel coils are produced on rolling mills by deforming rectangular pieces of steel referred to as slabs. These slabs are the result of a continuous casting machine transforming batches (as much as 300 tons per batch) of liquid steel into solid rectangular pieces of steel, that is, slabs.

The benefits of a semi-finished inventory to act as a buffer between the steelmaking–continuous casting operations and the rolling mills are well known in the steel industry.[6–9] The selection of what sizes to stock is an important combinatorial optimization problem that can be formulated as a PCUFLP. It is interesting to note that the following application has nothing at all to do with locating facilities, but, in reality, is a type of inventory size consolidation problem.

The continuously cast slabs can be cut into several smaller slabs before being processed through the rolling mills. The problem was to determine, by metallurgical grade (steel recipe), the *minimum* number of continuously cast slab sizes (both width and length—thickness was known) such that 85 to 90% of the customer order requirements (by weight) could be met from this semi-finished inventory of slabs. The remaining 10 to 15% of the customer requirements would be met from made-to-order steel production.

The 10 to 15% of the customer requirements not provided from the semi-finished inventory consisted of small quantity orders that were more economically made to order. It is precisely the fact that less than 100% of the customer requirements needed to be met from the semi-finished inventory that motivated the introduction of the universal warehouse concept to accommodate the last 10 to 15% of the requirements, that is, the defining characteristic of a PCUFLP.

Determining if a given inventory size was feasible for a particular customer requirement required the following set of feasibility checks to be made:

1. Customer dimensional and weight specifications,
2. rolling mill processing constraints, and
3. continuous caster constraints.

If all these checks indicate feasibility, then the corresponding a_{ij} coefficient is set to one in the PCUFLP formulation. Otherwise, the a_{ij} value is set to zero.

Erlenkotter[5] developed a highly efficient dual ascent-based branch-and-bound algorithm for the UFLP called DUALOC. This FORTRAN code has been used by Nauss and Markland[10] as part of a procedure for lock box location. It has also been used by Pentico[11] and by Vasko and Wolf[12] as part of solution procedures to solve assortment problems. In addition, Vasko *et al*[9] used it as part of a procedure to determine as-cast bloom lengths, that is, the length of blooms produced directly from a caster.

The current application has similarities to the work described in Vasko and Wolf,[12] as well as Vasko *et al.*[9] In all three cases a separate problem is formulated for each metallurgical grade under consideration. Also, in each case, the candidate warehouses represent the potential semi-finished inventory sizes, and the retail stores represent the customer requirements.

However, there are two important differences:

1. First, the current problem size is much larger. In Vasko and Wolf[12] the largest UFLP had only 232 candidate warehouses and in Vasko *et al*[9] the largest UFLP had only 100 candidate warehouses.
2. Secondly, and more importantly, in the current problem the inventory sizes are to be designed to satisfy between 85 and 90% of the customer requirements (by weight). In the previous problems, 100% of the customer requirements had to be satisfied from inventory.

It is this second difference that we use to our advantage in formulating the problem as a PCUFLP.

In the current application, there are about 19 000 distinct slab sizes (candidate warehouses) that are potentially feasible for a given metallurgical grade. The actual number of feasible sizes (width by length) depends on the customer specifications within a metallurgical grade—not all of these sizes are needed for each grade. The largest number feasible for any given grade, based on annual customer requirements, is about 13 600 distinct slab sizes.

A retail store consists of a group of customer requirements that are identical based on customer specifications for metallurgical grade, chemistry requirements, dimensional requirements, and slab weight requirements. For a given metallurgical grade, the largest number of distinct retail stores is about 1650.

Feasible retail store–candidate warehouse pairs are determined by checking all customer dimensional and weight specifications, all rolling mill processing constraints, and all continuous caster constraints. In other words, all these checks (the specific constraints parameters are proprietary) boil down to determining whether the a_{ij} coefficients are 1 (feasible) or 0 (infeasible). For a given metallurgical grade, the largest number of feasible store–warehouse pairs is about 213 000.

Solutions with different numbers of distinct slab sizes can be generated by varying the fixed cost associated with each slab size. It should be noted that the same fixed cost is used for all slab sizes, and therefore no slab size is favored based on fixed cost. To model the fact that only 85 to 90% of the customer order requirements need to be satisfied from inventory, a universal slab size is added to the list of candidate warehouses. As discussed previously, this universal slab size has a zero fixed cost and every customer requirement can be met from it. It is important to realize that in this PCUFLP application the fixed costs do *not* represent actual costs, but instead, these costs are modelling means for controlling the amount of customer requirements (by tons) that are assigned to actual slab sizes in the solution. The higher the fixed costs, the lower the percentage of customer requirements satisfied from inventory. The lower the fixed costs, the higher the percentage of customer requirements satisfied from inventory.

The variable costs are defined, primarily, so that, for a fixed number of inventory slab sizes in the PCUFLP solution, the customer requirements by weight will be maximized (tons not satisfied from inventory replace lost sales). For a given PCUFLP solution, customer order requirements assigned to the universal slab size represent material to be produced on a made-to-order basis instead of being supplied from inventory.

Semi-finished inventory size results

For this application, semi-finished inventory sizes were designed and implemented for about 30 different metallurgical grades. In this section, we analyse, in some depth, results from three of the metallurgical grades. For proprietary reasons, these three grades will simply be referred to as small, medium, and large based on their corresponding PCUFLP problem size. Summary information for these three grades at six different fixed cost values are given in Tables 1 and 2. Each of the three grades was executed at all six fixed cost levels to illustrate solution sensitivity to fixed cost. These tables represent a total of 18 PCUFLPs. The number of rows refers to the number of retail stores in a problem. The number of columns refers to the number of candidate warehouses in a problem and the number of feasible pairs refers to the number of feasible retail store–candidate warehouse pairs. Table 1 gives the total number of dual ascent iterations until the program terminates with the guaranteed optimal solution, the number of iterations at which the optimal solution was found, and the iteration at which the branch-and-bound strategy was initiated. Table 2 gives the total number (including the universal warehouse) of warehouses in the solution, the percentage of customer weight requirements met by the solution and the average percentage of customer requirements met per warehouse. Executed on a 500 MHz PC, about 275 dual ascents were performed per second.

For this application the desired solution (85 to 90% coverage) occurred at a different fixed cost for each of the three grades. For a given metallurgical grade, that is, problem size, the strategy is to start with a high fixed cost and to decrease it until between 85 and 90% of the customer requirements can be satisfied from the semi-finished inventory. As seen from Tables 1 and 2, for the small problem, the desired solution was found in less than 3 s at a fixed cost of 12 500 with 88% of the customer weight supplied from 15 semi-finished inventory sizes. For the medium problem, the desired solution was found in about 90 s at a fixed cost of 25 000 with 88% of the customer weight supplied from 17 semi-finished inventory sizes. For the large problem, the desired solution was found in about 30 s at a fixed cost of 50 000 with 89% of the customer weight supplied from 20 semi-finished inventory sizes.

Even if we needed to cover as much as 95% of the customer order weight requirements for the small and medium

Table 1 Iteration results for structured data

Fixed cost	*Small (269 rows, 6036 columns, 31 904 feasible pairs)*	*Medium (929 rows, 10 625 columns, 117 504 feasible pairs)*	*Large (1649 rows, 13 552 columns, 212 870 feasible pairs)*
400 000	1	151	2035
	1	151	1688
	NA	NA	NA
200 000	1	720	3076
	1	689	3073
	NA	NA	NA
100 000	1	1174	4460
	1	1171	4460
	NA	NA	NA
50 000	157	3970	8510
	157	3970	8510
	NA	3250	7172
25 000	254	25 056	4 469 425
	254	22 962	4 469 383
	NA	4521	36 438
12 500	754	130 479	5 843 403
	754	115 031	5 832 574
	NA	11 516	40 507

problems, optimal solutions would be achievable in a few seconds (small) to a few minutes (medium). However, if we needed to cover as much as 95% for the large problem and wanted to guarantee the optimal solution, then it would take over 4 h. As will be shown later, for this type of application, with high empirical confidence of being within 1% of optimum, DUALOC can be terminated after, at most, 500 000 dual ascent iterations (about 30 min on a 500 MHz PC).

For this particular application, the goal was to optimally satisfy 85 to 90% of the customer weight requirements from the minimum number of inventory sizes. Each metallurgical grade required the solution of no more than five PCUFLPs in order to achieve the correct coverage. Also, each PCUFLP was solved to optimality in two minutes or less.

Analysis of DUALOC performance

As practitioners we were very pleased that DUALOC was capable of generating optimal solutions to these large applications so efficiently. We were curious as to what structure in our problems allowed this classic algorithm,

Table 2 Optimal PCUFLP results for structured data

Fixed cost	*Small (269 rows, 6036 columns, 31 904 feasible pairs)*	*Medium (929 rows, 10 625 columns, 117 504 feasible pairs)*	*Large (1649 rows, 13 552 columns, 212 870 feasible pairs)*
400 000	1	2	6
	0%	24%	59%
	0%	12%	9.8%
200 000	1	5	10
	0%	52%	74%
	0%	10.4%	7.4%
100 000	1	8	15
	0%	68%	83%
	0%	8.5%	5.5%
50 000	4	13	21
	42%	81%	89%
	10.5%	6.2%	4.2%
25 000	8	18	34
	66%	88%	96%
	8.3%	4.9%	2.8%
12 500	16	27	46
	88%	94%	98%
	5.5%	3.5%	2.1%

coded in FORTRAN back in 1976 by Donald Erlenkotter,[5] to perform so well. Thus, we decided to systematically vary the input of these three problems.

1. We rearranged the candidate warehouses (the candidate warehouses were originally sorted in a certain order by size). Alternative sorting criteria had no impact on solution time.
2. We rearranged the candidate warehouses randomly, but this also did not change the solution time.
3. We kept the *number* of candidate warehouses that were feasible for each retail store the same, but, using a separate random uniform distribution for each retail store, assigned feasible candidate warehouses to each store. This kept the overall density of the problem constant and it kept the number of feasible candidate warehouses per retail stores the same, but it changed the distribution of number of retail stores satisfied by a candidate warehouse. In other words, the average number of retail stores that a warehouse could service remained the same, but the number of retail stores that any particular warehouse could service is now different.

In summary, cases 1 and 2 tested the impact of column sorting on solution time. Because of the input format of DUALOC, it was not surprising that these two cases had no impact on solution time. Specifically, the DUALOC input requires that, for each retail store, the candidate warehouses for that store be listed in ascending order based on the variable cost of meeting that store's demand. In other words, no matter how the candidate warehouses (columns) are sorted or labeled in the model, the input to DUALOC remains the same. In case 3, the impact of the distribution of number of retail stores satisfied by a candidate warehouse was determined. The same 18 PCUFLPs were solved using this randomized data for the small, medium, and large problems. The results are summarized in Tables 3 and 4.

With this 'randomized' data the number of dual ascent iterations required to solve these problems has increased significantly. Only for relatively high fixed cost values could these problems now be solved to optimality. The DUALOC program was terminated after 10 million dual ascent iterations (approximately 10 h of execution time). The smallest fixed costs in the tables for which each problem was solved to optimality by DUALOC were 25 000 for the small problem with 63% of the customer weight requirements covered by eight (nine with the universal warehouse) semi-finished inventory sizes, 50 000 for the medium problem with 61% of the customer weight requirements covered by 15 (16 with the universal warehouse) semi-finished inventory sizes, and 200 000 for the large problem with only 44% of the customer weight requirements covered by eight (nine with the universal warehouse) semi-finished inventory sizes.

As a first step in what was to be a detailed fitness landscape analysis,[13,14] the distributions (random and structured) of the number of retail stores satisfied per candidate warehouse were plotted on graphs (Figures 1, 2, and 3) for each of the three problems. Specifically, for each of the three problem sizes and two input data sets (random and structured), we determined the number of candidate warehouses that could satisfy exactly one retail store, exactly two retail stores, exactly three retail stores, and so on. This information was then organized as ordered pairs (num_rs, num_cw), in which num_rs was the number of retail stores satisfied and num_cw was the number of candidate warehouses that

Table 3 Iteration results for random data

Fixed cost	*Small (269 rows, 6036 columns, 31 904 feasible pairs)*	*Medium (929 rows, 10 625 columns, 117 504 feasible pairs)*	*Large (1649 rows, 13 552 columns, 212 870 feasible pairs)*
400 000	1	1	6
	1	1	6
	NA	NA	NA
200 000	1	4	15 854
	1	4	15 853
	NA	NA	1158
100 000	1	313	10 000 000[a]
	1	313	9 893 483
	NA	233	20 928
50 000	151	2 791 620	10 000 000[a]
	150	2 654 135	8 813 362
	118	4201	136 761
25 000	8735	10 000 000[a]	10 000 000[a]
	8020	864 867	8 126 509
	553	40 237	1 303 636
12 500	10 000 000[a]	10 000 000[a]	10 000 000[a]
	114 716	921 991	7 635 222
	4642	169 170	1 038 891

[a]Maximum 10 000 000 iterations exceeded.

Table 4 Optimal PCUFLP results for random data

Fixed cost	*Small (269 rows, 6036 columns, 31 904 feasible pairs)*	*Medium (929 rows, 10 625 columns, 117 504 feasible pairs)*	*Large (1649 rows, 13 552 columns, 212 870 feasible pairs)*
400 000	1	1	1
	0%	0%	0%
	0%	0%	0%
200 000	1	1	9
	0%	0%	44%
	0%	0%	4.9%
100 000	1	6	19
	0%	33%	56%
	0%	5.5%	2.9%
50 000	4	16	63
	37%	61%	86%
	9.3%	3.8%	1.4%
25 000	9	55	118
	63%	90%	97%
	7%	1.6%	0.8%
12 500	25	86	155
	88%	98%	99%
	3.5%	1.1%	0.6%

satisfied num_rs retail stores. For example, in Figure 1, the left-most data point and the right-most data point of the structured plot are (1, 1678) and (24, 11), respectively. For the small problem using the actual (structured) input data, there were 1678 candidate warehouses that could service exactly one retail store and there were 11 candidate warehouses that could service exactly 24 retail stores. In contrast, for the randomized input data for this problem, the left-most data point and the right-most data point were (1, 154) and (15, 3), respectively. That is, when the input was randomized for the small problem, there were only 154 candidate warehouses that could service exactly one retail store and there were 3 candidate warehouses that could service exactly 15 retail stores. For the randomized input data, 15 stores were the most that any candidate warehouse could service versus 24 for the actual problem data.

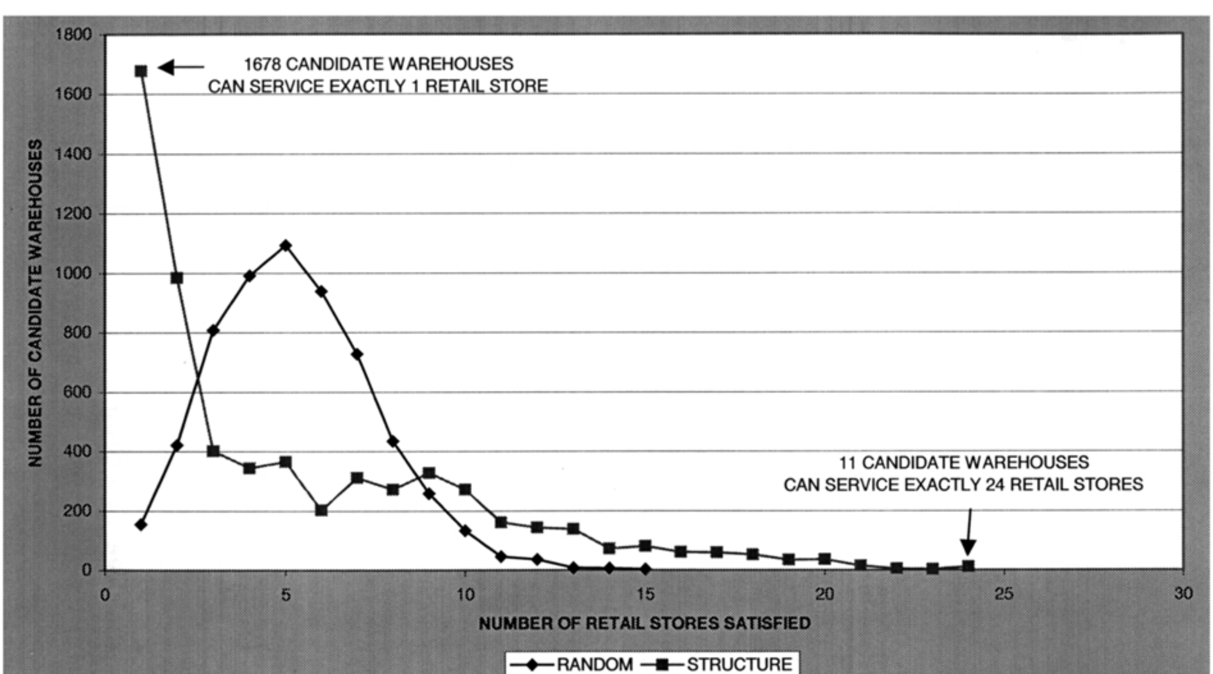

Figure 1 Number of candidate warehouses *vs* number of retail stores satisfied (small problem).

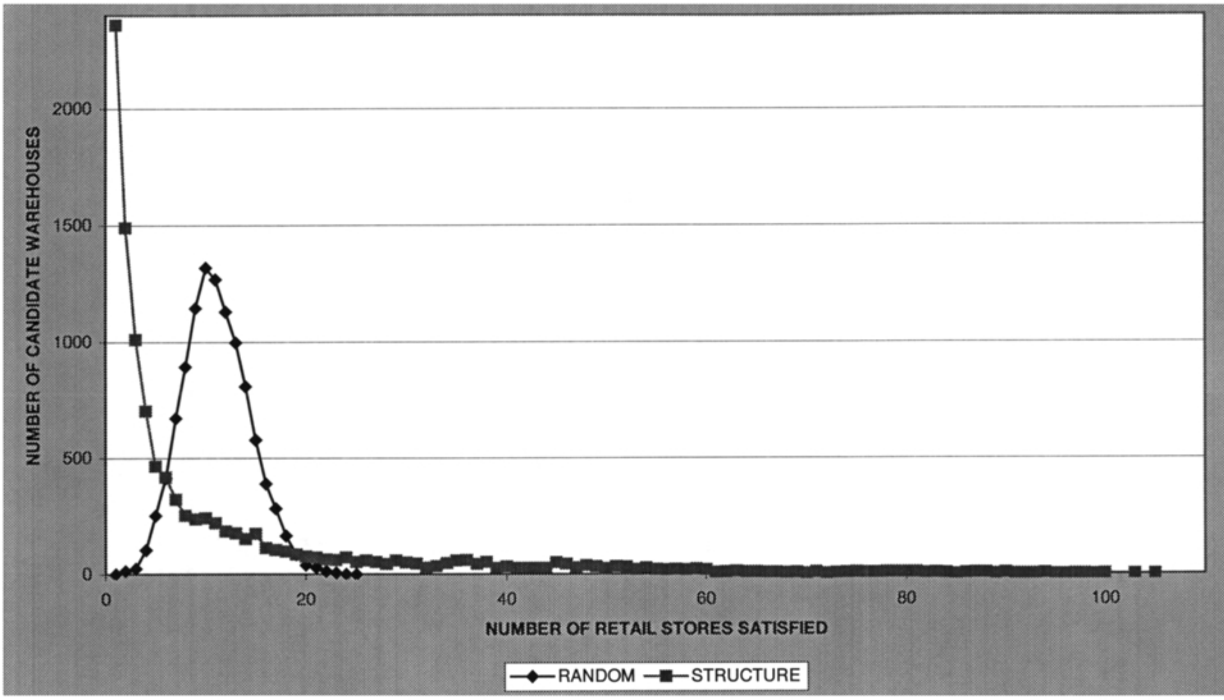

Figure 2 Number of candidate warehouses *vs* number of retail stores satisfied (medium problem).

As can be seen in Figures 1, 2, and 3, the distributions for the random versus the structured inputs are vastly different; however, the *average* number of retail stores satisfied per candidate warehouse is the same for each problem input. For the actual (structured) problems, the data are approximately exponentially distributed. For the randomized data problems, the data are normally distributed, as expected, since they are the sum of uniform random distributions.

For the structured (exponential distributions, there are a few candidate warehouses that satisfy many retail stores (the right tail of the distributions). Clearly, PCUFLP solutions should be comprised heavily of these candidate warehouses,

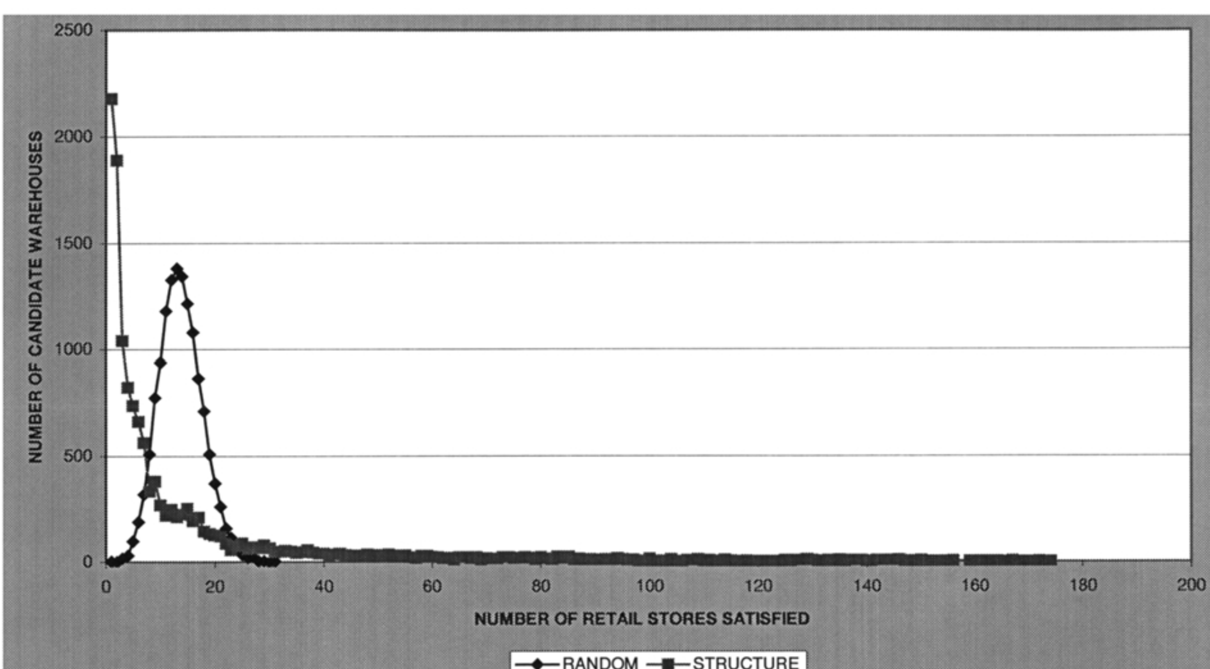

Figure 3 Number of candidate warehouses *vs* number of retail stores satisfied (large problem).

which are easy for DUALOC to identify because there are few in number. Also, for the exponential distributions, there are a large number of candidate warehouses that satisfy very few retail stores (the left tail of the distributions). Obviously, these candidate warehouses should not be in PCUFLP solutions.

In contrast to the exponential distributions of the actual real-world applications, the randomized data problems result in normal distributions. These normal distributions do not provide the same easy identification of which candidate warehouses should be in a solution and which should not be in a solution, hence, the much longer execution times for the randomized data problems.

Also, Figures 1, 2, and 3 can be used to explain the relationship between execution time required to solve the problem and fixed cost. That is, for high fixed costs, the warehouses are chosen from the extreme right tail of the distributions. This is easy to do regardless of the distribution. However, as the fixed costs are reduced, allowing more warehouses into the solution, the PCUFLP solution is generated by starting at the right tail of the distribution and 'moving' to the left. The lower the fixed costs, the further to the left DUALOC has to search for the optimal solution.

DUALOC as a heuristic

Now let us consider using DUALOC heuristically in order to solve large PCUFLPs. Suppose, for example, the large problem application needed to be solved for a customer weight coverage of 95 to 100%, and for computational efficiency we needed to limit execution time to, say, 30 min, which is equivalent to about 500 000 dual ascent iterations on a 500 MHz PC. In order to solve such problems, we could consider a heuristic specifically designed for the UFLP[15,16] or we could make use of a general heuristic or metaheuristic strategy[17,18] that we would need to customize to solve our particular PCUFLPs.

However, given that DUALOC is efficiently alternating between improved dual solutions and corresponding primal solutions,[5] it appeared that DUALOC could also be used heuristically. Specifically, DUALOC could be terminated (without proving optimality) after either a predetermined number of dual ascents, or when solution improvement 'slowed down', or both. The convergence of DUALOC for a fixed cost of 12 500, for both actual data and random data, is illustrated in Figures 4, 5, and 6 for the small, medium, and large problems, respectively.

For the *actual* applications, even at a fixed cost of 12 500, the small (Figure 4) and medium problems (Figure 5) are solved to optimality in less than three seconds and less than eight minutes, respectively. For the large problem (98% coverage from inventory), the optimal solution is not found until dual ascent iteration 5 832 574 with optimality proven shortly after iteration 5 843 403 (just under 6 h of execution time). From Figure 6, we see that DUALOC has found a solution with 1.2 and 0.6% of optimality at about 100 000 and 300 000 dual ascent iterations, respectively. Based on the three applications discussed in this paper and numerous other semi-finished inventory design problems (other metallurgical grades) that we were called upon to determine sizes for, very good solutions (within

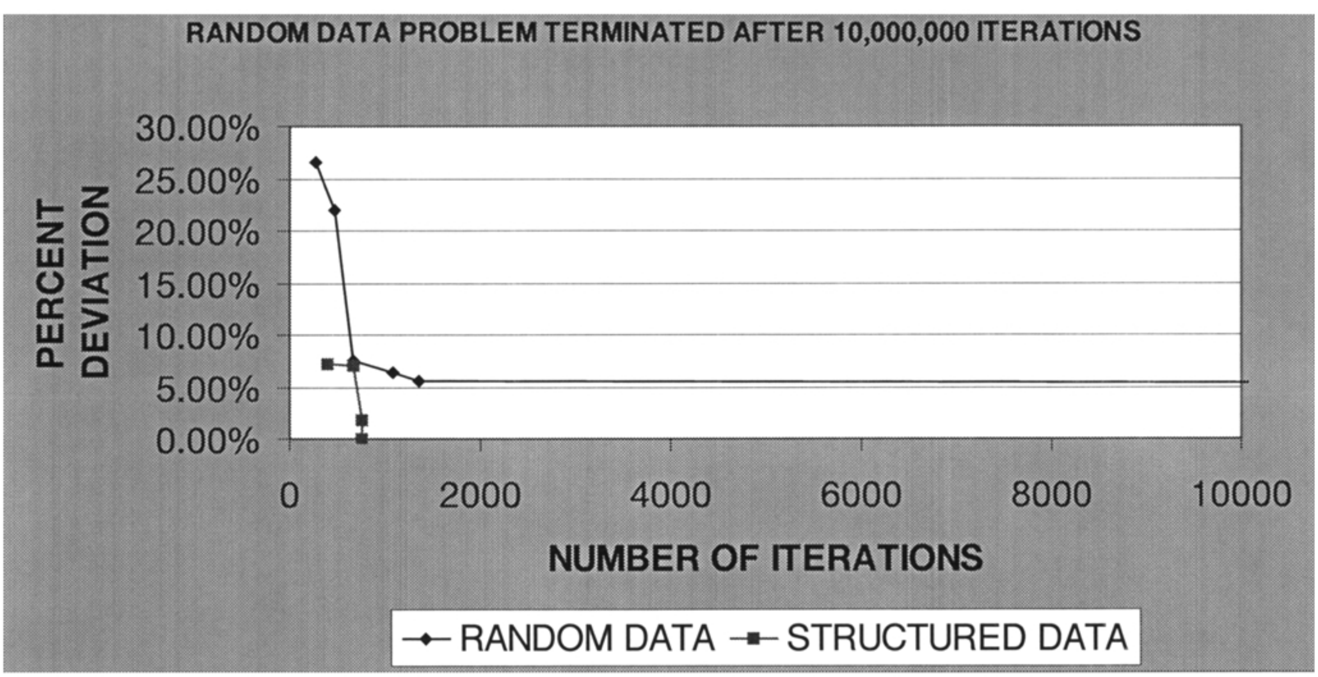

Figure 4 Percent deviation from optimum *vs* number of iterations (small problem).

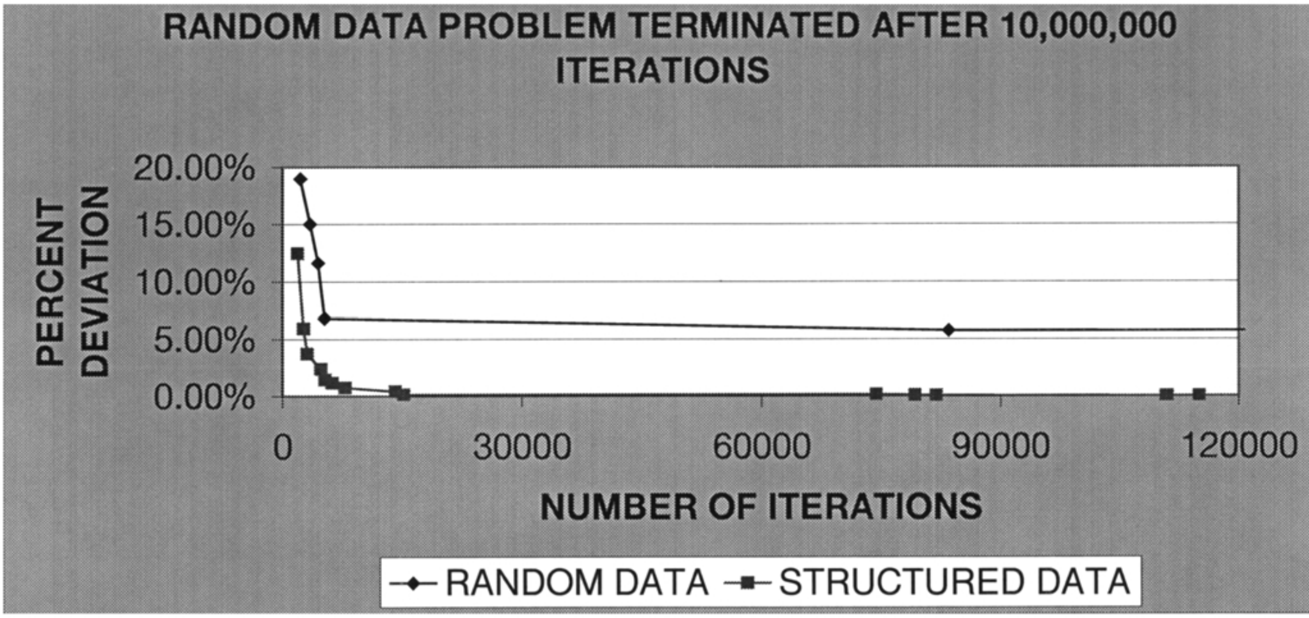

Figure 5 Percent deviation from optimum *vs* number of iterations (medium problem).

1.0% of optimality) for *any* of these problems at *any* fixed cost level could be achieved by executing DUALOC. This was true for cases requiring at most 500 000 iterations (about 30 min on a 500 MHz PC).

In contrast, for the *randomized* data problems, DUALOC converged much more slowly. At a fixed cost of 12 500, all three of the problems were terminated after 10 million dual ascent iterations. For the small problem, the best PCUFLP solution was found by iteration 114 716, but optimality was still not proven after 10 million iterations. For the medium problem, the best PCUFLP solution was found by iteration 921 991, but optimality was still not proven after 10 million iterations. For the large problem, the best PCUFLP solution was not found until iteration 7 635 222, and was still trying to prove optimality after 10 million iterations. For these randomized data problems, where the

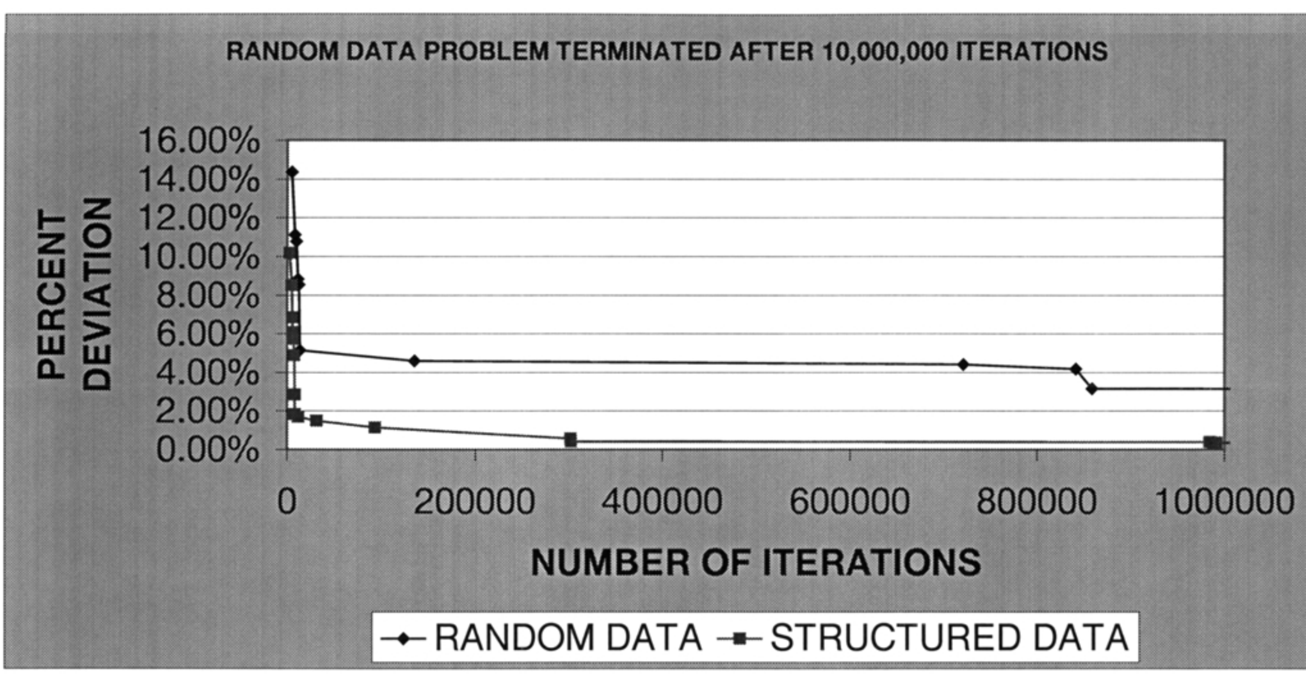

Figure 6 Percent deviation from optimum *vs* number of iterations (large problem).

distribution of the number of retail stores satisfied per warehouse does not allow for the easy identification of which candidate warehouses should or should not be in the solution, we would expect any algorithm to perform poorly. In light of this fact, it is a credit to DUALOC that for these problems, it converges to within 5% of the best-known solution after, at most, 500 000 dual ascent iterations.

Conclusions

The PCUFLP is a generalization of the UFLP, in which not all retail stores need to have their demand satisfied from a warehouse. The problem of determining semi-finished inventory sizes to be used to produce sheet steel products was formulated as a PCUFLP. Erlenkotter's classic dual-ascent algorithm, DUALOC, was used to solve real-world PCUFLP applications with over 1600 retail stores, 13 000 candidate warehouses, and about 213 000 feasible retail store–candidate warehouse pairs in less than two minutes on a 500 MHz PC.

The ability to obtain efficient solutions to large semi-finished inventory size applications formulated as PCUFLPs is dependent on two key factors: (1) the percentage of customer weight requirements that are to be met from the semi-finished inventory; and (2) how the number of retail stores satisfied per candidate warehouse are distributed. For large inventory size problems, as the percentage of customer order weight covered by inventory increases, that is, the fixed costs decrease, the problem becomes more difficult to solve.

For general PCUFLPs, efficient solution times can be attributed to the fact that the number of retail stores satisfied by a candidate warehouse is approximately exponentially distributed. In contrast, if the number of retail stores satisfied by a candidate warehouse is approximately normally distributed, then large instances of the corresponding PCUFLP are expected to be difficult to solve. Both of these results will hold true regardless of the particular application. This characterization of the distribution of number of retail stores satisfied by a candidate warehouse can be extremely helpful to any practioner in deciding on how to utilize the classic DUALOC algorithm as either an exact or an approximate solution procedure.

Appendix

Outline of DUALOC logic

A dual ascent procedure is used efficiently to generate solutions to a condensed dual of the UFLP. Complementary slackness relationships for optimal linear programming solutions are used to derive feasible primal solutions from dual solutions. If the complementary slackness conditions are satisfied, then the optimal solution has been obtained. Otherwise, a dual adjustment procedure is executed in order to try to reduce complementary slackness violations. If iterative use of the dual ascent and adjustment procedures do not yield an optimal integer solution to the UFLP, then a branch-and-bound procedure is executed.

References

1 Daskin MS (1995). *Network and Discrete Location: Models, Algorithms, and Applications.* Wiley: New York.
2 Re Velle CS and Laporte G (1996). The plant location problem: new models and research prospects. *Opns Res* **44**: 864–874.
3 Gourdin E, Labbe M and Laporte G (2000). The uncapacitated facility location problem with client matching. *Opns Res* **48**: 671–685.
4 Daskin MS and Owen SH (1999). Two new location covering problems: the partial covering p-center problem and the partial set covering problem. *Geographical Anal* **31**: 217–235.
5 Erlenkotter D (1978). A dual-based procedure for uncapacitated facility location. *Opns Res* **26**: 992–1009.
6 Cowling P and Rezig W (2000). Integration of continuous caster and hot strip mill planning for steel production. *J Scheduling* **3**: 185–208.
7 Kalagnamam J, Dawande MW, Trumbo M and Lee HS (2000). The surplus inventory matching problem in the process industry. *Opns Res* **48**: 505–516.
8 Newhart DD, Stott KL and Vasko FJ (1993). Consolidating product sizes to minimize inventory levels for a multi-stage production and distribution system. *J Opl Res Soc* **44**: 637–644.
9 Vasko FJ, Newhart DD, Stott KL and Wolf FE (1996). Using a facility location algorithm to determine optimum cast bloom lengths. *J Opl Res Soc* **47**: 341–346.
10 Nauss RM and Markland RE (1981). Theory and application of an optimizing procedure for lock box location analysis. *Mngt Sci* **27**: 855–865.
11 Pentico DW (1988). The discrete two-dimensional assortment problem. *Opns Res* **36**: 324–332.
12 Vasko FJ and Wolf FE (1994). A practical approach for determining rectangular stock sizes. *J Opl Res Soc* **43**: 691–706.
13 Reeves CR (1999). Landscapes, operators and heuristic search. *Ann Opl Res* **86**: 473–490.
14 Merz P and Freisleben B (2000). Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE Trans Evolutionary Comput* **4**: 337–352.
15 Shmoys DB, Tardos E and Aardal K (1997). Approximation algorithms for facility location problems. In: Proceedings of 29th ACM Symposium on Theory of Computing. Association for Computing Machinery, New York, pp. 265–274.
16 Barahona F and Chudak FA (1999). Solving large scale uncapacitated facility location problems. In: Pardalos PM (ed). *Approximation and Computing in Numerical Optimization: Continuous and Discrete Problems.* Kluwer Academic Publishers: Boston, MA.
17 Michalewicz Z and Fogel DB (2000). *How to Solve It: Modern Heuristics*, Springer-Verlag: New York, NY.
18 Voss S, Martello S, Osman IH and Roucairol C (eds) (1999). *META-Heuristics: Advances and Trends in Local Search Paradigms for Optimization.* Kluwer Academic Publishers: Boston, MA.

Received September 2001;
accepted June 2002 after two revisions