MANUFACTURING & SERVICE OPERATIONS MANAGEMENT
Vol. 7, No. 3, Summer 2005, pp. 188–207
ISSN 1523-4614 | EISSN 1526-5498 | 05 | 0703 | 0188

DOI 10.1287/msom.1050.0083
© 2005 INFORMS

Trade-offs Between Customer Service and Cost in Integrated Supply Chain Design

Zuo-Jun Max Shen
Department of Industrial Engineering and Operations Research, University of California, Berkeley, California 94720-1777, shen@ieor.berkeley.edu

Mark S. Daskin
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois 60208, m-daskin@northwestern.edu

When designing supply chains, firms are often faced with the competing demands of improved customer service and reduced cost. We extend a cost-based location-inventory model (Shen et al. 2003) to include a customer service element and develop practical methods for quick and meaningful evaluation of cost/service trade-offs. Service is measured by the fraction of all demands that are located within an exogenously specified distance of the assigned distribution center. The nonlinear model simultaneously determines distribution center locations and the assignment of demand nodes to distribution centers to optimize the cost and service objectives. We use a weighting method to find all supported points on the trade-off curve. We also propose a heuristic solution approach based on genetic algorithms that can generate optimal or close-to-optimal solutions in a much shorter time compared to the weighting method. Our results suggest that significant service improvements can be achieved relative to the minimum cost solution at a relatively small incremental cost.

Key words: customer service; integrated supply chain design; multiobjective optimization; trade-off analysis; genetic algorithm
History: Received: September 5, 2003; accepted: June 30, 2005. This paper was with the authors 10 months for 3 revisions.

1. Introduction

Strategic supply chain design and redesign have become a major challenge for firms as they simultaneously try to improve customer service and reduce operating costs. Three major cost factors associated with designing and managing a supply chain are the facility *location* costs, the *inventory* management costs, and the *distribution* costs. These three cost elements are highly related and, ideally, should be considered jointly when making supply chain design decisions. Although building a decision-support system that integrates these cost elements with customer service goals is a considerable undertaking for most businesses, doing so can provide a company with a tremendous competitive advantage in the marketplace.

Determining the number and location of distribution centers (DCs) is a critical task in the design of an effective supply chain network. The location theory literature that addresses this problem is extensive; however, the majority of these location models do not consider the inventory related costs. Only recently, Shen (2000), Shen et al. (2003), and Daskin et al. (2002) introduced a joint location-inventory model that incorporates working inventory and safety stock inventory costs at the DCs into the location model. Working inventory cost includes the fixed costs of placing orders, the shipment costs from the supplier to the DCs, and holding cost of the working inventory. The problem is to find the number and location of the DCs, and to assign customers to DCs, so as to minimize facility location costs, shipment costs, and inventory costs. The key difficulty in the model is that both the inventory costs at each DC and the shipment costs from the plant to the DCs depend nonlinearly on customer assignments, which are endogenously determined and are not known a priori. However, the pricing subproblem (in the case of column generation used by Shen 2000 and Shen et al. 2003) or the Lagrangian subproblem (in the case of Lagrangian relaxation as used by Daskin et al. 2002) can be solved efficiently in low order polynomial time, resulting in effective solution algorithms that can solve the overall problem for instances with hundreds of customer and candidate DC nodes in a few minutes. One key observation they make is that the traditional uncapacitated fixed charge (UFC) location model, which does not include any inventory costs, always underestimates the total cost and opens more DCs than does the location-inventory model.

An important supply chain design consideration that has been ignored in these papers is customer responsiveness. For instance, in the Shen et al. (2003) cost-based model, a retailer may be assigned to a DC that is very far away if doing so reduces the total costs. This may not be desirable in a highly competitive business environment. Many companies consider service time, defined as how long it takes to transport the products (services) to the customer site when they are needed, to be a critical performance metric. For example, General Motors Corporation developed a program in Florida to reduce the amount of time that Cadillac buyers wait for new cars. New cars are held at a DC and are made available to the dealers on order. A 24-hour delivery standard was used to deliver new cars from DCs to dealers (Stern 1995). This program has since been expanded to include areas in Maryland and California. Another example comes from the PC industry. To provide high-quality service, IBM has implemented a parts stocking plan to support a time-based service strategy. Specifically, IBM wants to keep the response time within a threshold level, say 2 hours to one set of customers, 8 hours to another set of customers, and 24 hours for the rest of the customers (Ma and Wilson 2002).

In both of the above environments, determining the locations of the DCs that will hold the inventory (e.g., vehicles, parts) is critical, because it will impact both the total costs (facility location costs, inventory costs, and transportation costs) and the customer responsiveness. There is a clear need to evaluate trade-offs among the total costs and customer service. In this paper, we incorporate customer responsiveness into a supply chain design model and analyze such trade-offs.

The rest of this paper is organized as follows. Section 2 discusses relevant results in the literature. Section 3 provides the formulation of the model, and §4 explains how to generate the trade-off between cost and service using the weighting method. Section 5 proposes a genetic algorithm approach to generating the trade-off curve. Computational results are provided in §6. Finally, §7 concludes this paper with directions for future research.

2. Literature Review

There exists an extensive literature on the multiobjective analysis of location problems; however, most of these papers focus on continuous and network models. We review only multiobjective discrete location models in this paper.

Ross and Soland (1980), in one of the earliest papers on multiobjective location problems, argue that practical problems involving the location of public facilities should be modeled as multiobjective problems. Cost and service are the typical objectives, although there exist several distinct objectives in each of those two categories: fixed investment cost, fixed operating cost, variable operating cost, total operating cost, and total discounted cost are all reasonable cost objectives to consider; and both demand served and response time (or distance traveled) are appropriate objectives for service measurement. They treat such multiobjective questions in the framework of a model for selecting a subset of M sites at which to establish public facilities to serve client groups located at N distinct points. They use an interactive approach that involves solving a finite sequence of the generalized assignment problems. Lee et al. (1981) apply an integer goal programming technique to a facility location problem with multiple cost-related objectives. The same technique has been applied to a multiobjective fire station location problem by Badri et al. (1998), who consider objectives that incorporate costs, travel times, and travel distances from stations to demand sites. Another technique, hierarchical programming, has also been applied to multiobjective location models where a clear ranking or hierarchy of the different objectives exists. Daskin and Stern (1981) address two hierarchical objectives in their paper on emergency medical service vehicle deployment. The primary objective is to minimize the number of service vehicles needed to satisfy a certain service requirement; and the secondary objective is to maximize the extent of multiple coverage of service zones. There also exist works that apply general multiobjective mixed-integer linear programming techniques to location problems. For example, Solanki (1991) develops an algorithm for solving biobjective mixed-integer linear programming problems and then applies it to biobjective location problems. This algorithm uses an approximate scheme to represent the noninferior solutions. Finally, simulation has also been used to validate multiobjective location models (Heller et al. 1989).

Multiobjective programming has also been applied to dynamic location problems (Schilling 1980) and location problems with uncertainty (Belardo et al. 1984). Belardo et al. study the problem of locating oil spill response equipment using a partial set covering model. With the existing equipment, their objective is to attain the best protection while minimizing the risk of being unprepared for events like oil spills. Other earlier works on multiobjective location models include a decision-support computer system (Hultz et al. 1981) and surveys (ReVelle et al. 1981, Current et al. 1990).

In a more general model, Jayaraman (1999) studies a multiobjective model for a service facility location problem with multiple products. Three objectives are considered in his model: (1) minimizing the fixed cost of opening facilities, (2) minimizing the total variable costs of serving customer demands, and (3) minimizing the average response time for serving customers.

Recently, Fernandez and Puerto (2003) consider the multiobjective uncapacitated facility location problem. Each objective corresponds to a different scenario proposed by a different decision maker on the realizations of facility setup costs and the allocation costs of customers to facilities. They also provide a review of other multiobjective location models, including continuous and network location models.

There are several papers that apply a multiobjective approach to supply chain planning models. For example, Sabri and Beamon (2000) propose an integrated multiobjective supply chain model for use in simultaneous strategic and operational supply chain planning. The objectives include cost, customer service levels (fill rates), and flexibility (volume or delivery). They tested their mathematical programming model on an example system consisting of three raw materials, two finished products, five vendors, three plants, four DCs, and five customer zones.

Nozick and Turnquist (2001) present an optimization model that is closely related to ours: minimize cost and maximize service coverage. They adapt the simple $(S-1, S)$ inventory policy and use a linear function to approximate the safety stock inventory cost function, which is then embedded in a fixed-charge facility location model. Using a linear function to approximate a nonlinear function makes the corresponding model much simpler; however, the error induced can be significant unless the number of DCs is very large. Furthermore, in their regression model, they need to recalibrate their model (the contribution to the fixed cost due to inventory) in every case because there is no clear relationship between the underlying costs (e.g., holding costs, fixed order costs, fixed transportation costs) and the fixed component of the regression they fit. The Nozick and Turnquist model is further simplified by ignoring some of the costs we consider (e.g., fixed order costs, fixed transportation costs from a plant to the DCs). Finally, they solve the problem heuristically.

In this paper, we propose a more realistic model by adding a service consideration into the supply chain design model proposed by Shen et al. (2003). As a result, we have a multiobjective optimization model that includes the fixed order costs, the fixed transportation costs from a plant to the DCs, and the nonlinear working inventory cost and the nonlinear safety stock inventory cost, resulting from a (Q, r) inventory policy with a specified cycle service level requirement. Furthermore, two solution approaches, one based on the weighting method and the other based on genetic algorithms, are proposed to solve this multiobjective model. The genetic algorithm performs very well compared to the weighting method, and it is the only feasible approach for large-sized problem instances, because the weighting method requires excessive computational time in these cases.

3. Formulation of the Model

In this section, we formulate a multiobjective model that can be used to explore the trade-off between the location, transportation, and inventory costs on the one hand and the level of service provided to the customers on the other. We explore this trade-off in the context of a multiechelon distribution system composed of a single plant, multiple DCs, and multiple retailers. The location-inventory model at the heart of this trade-off finds the optimal number and location of the DCs and the assignment of retailers to the DCs that minimize the sum of the (1) fixed DC location costs, (2) transportation costs from the plant to the DCs, (3) working inventory costs at the DCs, (4) safety stock costs at the DCs, and (5) transportation costs from the DCs to the retailers. Service is measured by the percentage of demand volume that can be served within an exogenously specified coverage distance.

We begin by summarizing the formulation of the location-inventory model on which our model is based on (Shen 2000, Daskin et al. 2002, Shen et al. 2003). We define the following notation:

$\mathbf{I}$	set of retailers,
$\mathbf{J}$	set of candidate DCs,
f_j	fixed (annual) cost of locating at candidate DC site j,
μ_i	mean daily demand at retailer i,
σ_i^2	variance of the daily demand at retailer i,
χ	days per year,
L_j	lead time (in days) to deliver from the plant to candidate site j,
d_{ij}	cost per unit shipped from candidate site j to retailer i,
h	holding cost per item per year (at the DC),
F_j	fixed administrative and handling cost of placing an order to the plant from candidate DC site j,
z_α	critical value of a standard normal random variable, Z, such that $P(Z > z_\alpha) = \alpha$,
$v_j(x)$	cost of shipping an order of size x from the plant to DC j which we later assume to be given by $v_j(x) = g_j + a_j x$, where we have:
g_j	fixed transportation cost per shipment and
a_j	cost per unit of a shipment from the plant to candidate site j.

To begin modeling the problem, assume for the moment that we know which customers are to be assigned to a specific DC. Assume that the demand at each retailer is Normally distributed with a daily mean of μ_i and a daily variance of σ_i^2, and the demands at each retailer are uncorrelated over time and across retailers. Let S be the set of customers assigned to the DC. Then, we let $D = \chi \sum_{i\in S} \mu_i$ be the total annual (expected) demand going through the DC. The safety stock required to ensure that stockouts during a lead time occur with a probability of α or less is $z_\alpha \sqrt{L \sum_{i\in S} \sigma_i^2}$.

Let $v_j(x)$ be the cost of shipping an order of size x from the plant to the DC. Then, the expected annual cost of ordering and holding working inventory at the DC is given by

$$F_j n + v_j\left(\frac{D}{n}\right) n + \frac{hD}{2n}, \tag{1}$$

where n is the (unknown) number of orders per year.

The first term of (1) represents the total fixed cost of placing n orders per year. The second term represents the shipment cost $v_j(D/n)$ per shipment multiplied by the number of shipments per year. D/n is the expected shipment size per shipment. The third term is the holding cost at the DC of the average working inventory. On average, there will be $D/(2n)$ items of working inventory on hand at a cost of h per item per year.

If $v_j(x)$ is linear in x, that is, $v_j(x) = g_j + a_j x$, where g_j denotes the fixed transportation cost per shipment and a_j denotes the cost per unit of a shipment from the plant to candidate site j, it is easy to show that $n = \sqrt{(hD)/[2(F_j + g_j)]}$ minimizes (1). By substituting this into the cost function (1), we obtain a working inventory cost of $\sqrt{2hD(F+g)} + aD$.

We also define the following decision variables as follows:

$$X_j = \begin{cases} 1 & \text{if we locate at candidate site } j, \\ 0 & \text{if not,} \end{cases}$$

$$Y_{ij} = \begin{cases} 1 & \text{if demands at retailer } i \text{ are assigned to a DC at candidate site } j, \\ 0 & \text{if not.} \end{cases}$$

Recall that D is not known a priori; rather it is given by $\chi \sum_{i\in\mathbf{I}} \mu_i Y_{ij}$ for any DC j.

With this notation, we can formulate an integrated location-inventory model as follows:

$$\text{Minimize} \quad \sum_{j\in\mathbf{J}} f_j X_j + \sum_{j\in\mathbf{J}}\sum_{i\in\mathbf{I}} \hat{d}_{ij} Y_{ij} + \sum_{j\in\mathbf{J}} K_j \sqrt{\sum_{i\in\mathbf{I}} \mu_i Y_{ij}} + \sum_{j\in\mathbf{J}} \Theta \sqrt{\sum_{i\in\mathbf{I}} \hat{\sigma}_{ij}^2 Y_{ij}} \tag{2}$$

$$\text{subject to} \quad \sum_{j\in\mathbf{J}} Y_{ij} = 1 \quad \forall\, i \in \mathbf{I}, \tag{3}$$

$$Y_{ij} \le X_j \quad \forall i \in \mathbf{I}, \forall j \in \mathbf{J}, \tag{4}$$

$$X_j \in \{0, 1\} \quad \forall j \in \mathbf{J}, \tag{5}$$

$$Y_{ij} \in \{0, 1\} \quad \forall i \in \mathbf{I}, \forall j \in \mathbf{J}, \tag{6}$$

where

$$\hat{d}_{ij} = \chi\mu_i(d_{ij} + a_j),$$
$$K_j = \sqrt{2h\chi}\sqrt{F_j + g_j},$$
$$\Theta = hZ_\alpha,$$
$$\hat{\sigma}_{ij}^2 = L_j\sigma_i^2.$$

The objective function (2) is composed of four terms. The first term represents the annual fixed cost of locating DCs. The second term captures the average annual cost of shipping goods from the DCs to the retailers, as well as the variable costs of shipping from the plant to the DCs. The third term represents the average annual working inventory costs and includes the fixed administrative and handling cost of ordering from the DC to the plant, as well as the fixed shipment costs from the plant to the DCs. Finally, the fourth term captures the average annual safety stock carrying costs. The first constraint (3) ensures that each demand node is assigned to exactly one DC. The second constraint (4) stipulates that demands can only be assigned to open DCs. Finally, constraints (5) and (6) are integrality constraints. Note that were it not for the final two terms of the objective function, the model would be structurally identical to an uncapacitated fixed charge location model.

The working inventory carrying costs represent the optimal economic order quantity costs, while the safety stock costs capture the cost of holding sufficient inventory to ensure that the probability of stocking out during a lead time is less than or equal to α. In essence, we are using a (Q, r) inventory model with a type I service-level requirement (Hopp and Spearman 1996, Nahmias 1997).

Finally, we assume that the ratio of the variance of the demand per unit time to the mean demand per unit time is the same for every retailer, that is, $\sigma_i^2/\mu_i = \gamma$ for every demand node i. While this may seem like an overly restrictive assumption, if the retailer demands are Poisson—and the Poisson distribution can be approximated very well by the normal distribution if the mean is sufficiently large—then this assumption is exact with $\gamma = 1$. With this additional assumption, we can combine the two nonlinear terms in (2) and rewrite the objective function as

$$\text{Minimize} \sum_{j\in\mathbf{J}} f_j X_j + \sum_{j\in\mathbf{J}}\sum_{i\in\mathbf{I}} \hat{d}_{ij} Y_{ij} + \sum_{j\in\mathbf{J}} \hat{K}_j \sqrt{\sum_{i\in\mathbf{I}} \mu_i Y_{ij}}, \tag{7}$$

where $\hat{K}_j = K_j + \Theta\sqrt{L_j\gamma}$.

This model is still a nonlinear integer programming model, but Daskin et al. (2002) show that the subproblems that result from relaxing constraint (3) can be solved in $O(|\mathbf{I}| \log |\mathbf{I}|)$ time for each such problem and there are $O(|\mathbf{J}|)$ such subproblems that must be solved at each Lagrangian iteration. When coupled with intelligent heuristics for finding good upper bounds and when the Lagrangian problem is embedded in a branch-and-bound solution approach, problems with hundreds of retailers and hundreds of candidate DCs can typically be solved in minutes on today's computers. Shu et al. (2005) report an efficient algorithm to solve the original problem (2)–(6) with two nonlinear terms in the objective function. To facilitate the computational work, we still assume that $\sigma_i^2/\mu_i = \gamma$ for every demand node i in this paper.

While the above model captures important facility location, transportation, and inventory costs, some retailers may be served very well, in the sense that they are located very close to the DCs to which they are assigned, while other retailers may be served very poorly by this criterion. The maximal covering location problem (Church and ReVelle 1974) maximizes the number of customers that can be covered by a fixed number of facilities. Customer i is covered if node i is assigned to a facility that is within d_c of node i, where d_c is the coverage distance. Instead of maximizing covered demand volume, for fixed total demand, we can minimize the uncovered demand volume. In a manner similar to Daskin (1995), we can then formulate a problem that simultaneously minimizes the fixed costs of the facilities and a weighted sum of the uncovered demand volume as follows:

$$\text{Minimize} \sum_{j\in\mathbf{J}} f_j X_j + W \sum_{j\in\mathbf{J}}\sum_{i\in\mathbf{I}} \tilde{d}_{ij} Y_{ij} \tag{8}$$

subject to (3)–(6),

where W is the weight on the uncovered demand volume and

$$\tilde{d}_{ij} = \begin{cases} \chi\mu_i & \text{if } d_{ij} > d_c, \\ 0 & \text{if not.} \end{cases}$$

Objectives (7) and (8) can be combined as follows:

$$\text{Minimize} \sum_{j\in\mathbf{J}} f_j X_j + \sum_{j\in\mathbf{J}}\sum_{i\in\mathbf{I}} \check{d}_{ij} Y_{ij} + \sum_{j\in\mathbf{J}} \hat{K}_j \sqrt{\sum_{i\in\mathbf{I}} \mu_i Y_{ij}}, \quad (9)$$

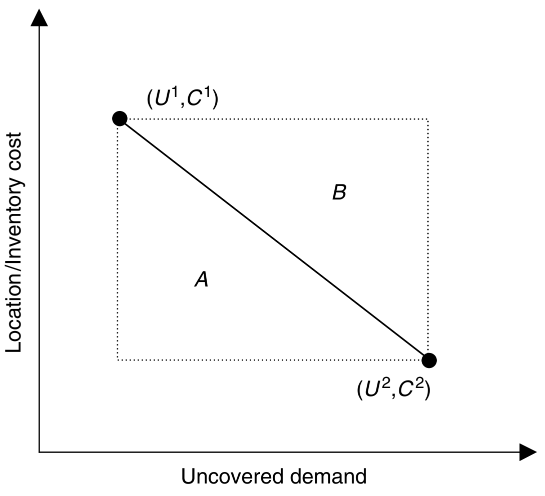
where $\check{d}_{ij} = \hat{d}_{ij} + W\tilde{d}_{ij}$. This model is structurally identical to (7). The only difference is that we penalize all assignments of demand nodes to DCs that are more than d_c away from the DC. By varying the weight W on uncovered demand volume, we can trace out an approximation to the set of noninferior solutions to the trade-off between location-inventory costs (7) and customer responsiveness. Very small values of W correspond to minimizing the total location-inventory cost; very large values of W are equivalent to minimizing the total location-inventory cost subject to the constraint that *all* demands are covered within a distance d_c of the DC to which they are assigned.

4. Generating the Trade-off Between Cost and Service Using the Weighting Method

With respect to a particular coverage distance d_c, we now illustrate how to determine appropriate values for W, the weight to be placed on uncovered demand volume to generate a trade-off curve. We define a solution with location-inventory costs C and uncovered demand volume U to be noninferior if there does not exist some other solution with cost C' and uncovered demand volume U' such that $C' \le C$ and $U' \le U$ with strict inequality holding for at least one of the two inequalities. Suppose that we already have two different noninferior solutions with location-inventory costs C^1 and C^2, respectively, and uncovered demands U^1 and U^2, respectively, as shown in Figure 1. The weighting method (Cohon 1978) attempts to find noninferior solutions to the south and west of the line connecting any two noninferior solutions when plotted in objective space. If we equate the objective function values at each of the two solutions, we obtain

$$C^1 + WU^1 = C^2 + WU^2$$

Figure 1 **Determining the Weight, W**



which, when solved for W, yields

$$W = \frac{C^2 - C^1}{U^1 - U^2}. \quad (10)$$

By selecting two adjacent noninferior solutions, we can use the weight determined in (10) in objective function (9) to determine whether or not there is a noninferior solution to the south and west of the line connecting the two solutions; i.e., in region A. The weighting method will not find noninferior solutions that might exist to the north and east of the line; i.e., in region B.

Determining the first two points to use in this process is relatively easy. To determine the bottom-most point, the one that minimizes the location-inventory cost, we simply minimize (7) subject to (3)–(6). For real-world systems there is unlikely to be a solution with equal (minimal) cost and smaller uncovered demand (i.e., directly to the left of the first point). For artificial data sets or for cases in which the analyst believes that such solutions might exist, objective function (7) can be minimized and the total location-inventory cost recorded. Then, objective function (9) can be minimized subject to (3)–(6) with a small weight on the uncovered demand volume. The total cost (excluding any penalties for uncovered demand volume) from this model can be compared to the total cost recorded for objective function (7) alone. If the two objective function values are the same, then the solution to (9) provides the optimal bottom-most solution in the objective space shown in Figure 1 (i.e., the bottom-most solution that is furthest to the left). If the costs are not the same (i.e., the cost from (9) exceeds that found using (7) alone), the weight W should be halved in the objective function (9) and the process repeated until the two costs are identical. In the computational results outlined below, we did not implement this process of halving the weight W because the data sets with which we are dealing are representative of real data for which the likelihood of there being an alternate optimal solution with smaller levels of uncovered demand is very small.

To find the left-most point of Figure 1, the solution that covers all demand at minimum cost, consider any demand node m that is not covered in the solution that minimizes cost without regard for coverage. The contribution of this node to the objective function (9) is

$$\chi\mu_m(d_{mj(m)}+a_{j(m)}+W)+\hat{K}_{j(m)}\left\{\sqrt{\sum_{i\in\mathbf{S}_{j(m)}}\mu_i}-\sqrt{\sum_{i\in\mathbf{S}_{j(m)}\setminus m}\mu_i}\right\},$$

where $j(m)$ is the index of the facility to which demand node m is assigned, W is the weight that will be assigned to uncovered demands, and $\mathbf{S}_{j(m)}$ is the set of demand nodes assigned to $j(m)$. One way for node m to be covered is to locate a facility at that node at a fixed cost of f_m. In that case, the contribution of node m to the objective function would be $f_m+\chi\mu_m(d_{mm}+a_m)+\hat{K}_m\sqrt{\mu_m}$. If this cost is less than the contribution of node m to the objective function when node m is assigned to $j(m)$, it will be advantageous to open a facility at m to serve node m. In other words, if

$$\begin{aligned}f_m+\chi\mu_m(d_{mm}+a_m)+\hat{K}_m\sqrt{\mu_m}&\\ <\chi\mu_m(d_{mj(m)}+a_{j(m)}+W)&\\ +\hat{K}_{j(m)}\left\{\sqrt{\sum_{i\in\mathbf{S}_{j(m)}}\mu_i}-\sqrt{\sum_{i\in\mathbf{S}_{j(m)}\setminus m}\mu_i}\right\}&,\end{aligned}$$

or if

$$\begin{aligned}W>\frac{1}{\chi\mu_m}&\left\{f_m+\hat{K}_m\sqrt{\mu_m}-\hat{K}_{j(m)}\left[\sqrt{\sum_{i\in\mathbf{S}_{j(m)}}\mu_i}-\sqrt{\sum_{i\in\mathbf{S}_{j(m)}\setminus m}\mu_i}\right]\right\}\\ &+\{d_{mm}+a_m-(d_{mj(m)}+a_{j(m)})\},\end{aligned}$$

it is better to open a facility at m to serve node m than to continue serving node m from $j(m)$. In our tests, all $\hat{K}_j$ are equal as are all a_j. Also, $d_{mm}=0$ for all nodes m. Thus, the condition above becomes

$$\begin{aligned}W>\frac{1}{\chi\mu_m}\Bigg\{f_m+\hat{K}_m\Bigg[\sqrt{\mu_m}-\sqrt{\sum_{i\in\mathbf{S}_{j(m)}}\mu_i}&\\ +\sqrt{\sum_{i\in\mathbf{S}_{j(m)}\setminus m}\mu_i}\Bigg]\Bigg\}&-d_{mj(m)}.\end{aligned}$$

Furthermore,

$$\begin{aligned}&\frac{1}{\chi\mu_m}\{f_m+\hat{K}_m\sqrt{\mu_m}\}-d_{mj(m)}\\ &>\frac{1}{\chi\mu_m}\left\{f_m+\hat{K}_m\left[\sqrt{\mu_m}-\sqrt{\sum_{i\in\mathbf{S}_{j(m)}}\mu_i}+\sqrt{\sum_{i\in\mathbf{S}_{j(m)}\setminus m}\mu_i}\right]\right\}-d_{mj(m)}\end{aligned}$$

(because $-\sqrt{\sum_{i\in\mathbf{S}_{j(m)}}\mu_i}+\sqrt{\sum_{i\in\mathbf{S}_{j(m)}\setminus m}\mu_i}<0$). The left-hand side of this inequality is easier to compute because it does not entail identifying the set $\mathbf{S}_{j(m)}$. Thus, to find the solution that covers all demand at minimal cost, we can set

$$W>\max_{m\in U}\left\{\frac{1}{\chi\mu_m}\{f_m+\hat{K}_m\sqrt{\mu_m}\}-d_{mj(m)}\right\},$$

where U is the set of demand nodes that are not covered in the solution to the pure minimum cost problem.

Once the two extreme solutions are known—the solution that minimizes the location-inventory cost and the solution that covers all demand at minimal cost—the procedure outlined above can be used iteratively until the solution to (9) for every pair of adjacent solutions in the solution space fails to identify a new point on the trade-off curve. Note that if n points on the final trade-off curve are identified (including the end points), we must solve $2n-1$ problems of the form of (9) subject to (3)–(6). Each of these problems is identical in structure to the model outlined in Daskin et al. (2002) and is solved using the algorithm outlined in that paper. In solving problems other than those needed to get the extreme points of the trade-off curve, however, we can provide an initial upper bound to the Lagrangian algorithm. In finding solutions that may lie between two solutions with coordinates in the objective space of (U^1, C^1) and (U^2, C^2), we know that $C^1+WU^1=C^2+WU^2$ is an upper bound on the objective function. In most of the results reported in the next section we used this bound and discuss the efficacy of this bound in §6.3. Qualitatively, this bound will affect the step size used at each iteration of the Lagrangian procedure. More importantly, it may allow us to prune certain nodes in the branch-and-bound tree earlier than we might be able to do in the absence of this bound.

As shown in the section on computational results, the exact algorithm outlined in this section requires excessive computational time for certain problem instances. One approach to resolving this problem is to solve the optimization problems approximately by considering a node in the branch-and-bound tree as pruned if the lower and upper bounds differ by less than ε percent, where $\varepsilon > 0$. In the next section, we discuss another approach based on a genetic algorithm.

5. A Genetic Algorithm Approach

In addition to implementing the solution technique outlined in §4, we developed a heuristic solution approach based on a genetic algorithm. Within the context of this approach, a solution is represented by a series of zeros and ones whose length is equal to the number of candidate DC nodes in the problem. This series corresponds to the location variables, X_j, defined earlier. In decoding a solution, we open facilities at the locations encoded as a one and assign every demand node to its nearest open facility. We do not attempt to improve the cost by exchanging assignments for demand nodes even though a solution that minimizes the combined location-inventory cost for given facility sites may entail assigning a demand node to a facility other than the nearest open site. We do this for three reasons. First, we want to keep the solution times relatively low. Second, the number of demand nodes that are assigned to a site other than the nearest site is often very small, as indicated in the computational results below. Finally, in our experience, the cost penalty that is paid for assigning demand volumes to the nearest facility as opposed to assigning them optimally is only on the order of a fraction of a percent. We note, however, that adding a heuristic to improve the assignment of customers to DCs may be important for systems whose parameters differ markedly from those we have tested. In particular, for systems with large variance to mean ratios for customer demands, the benefits of a more careful assignment of customers to DCs may be significant.

To evaluate the quality of any solution, we first find its cost and the demand volume that is left uncovered by the solution. Any solution k that is not dominated is assigned a score or value, V_k, of zero. Solutions that are dominated are compared to the nondominated solution that is to the immediate left and below the dominated solution in question. Dominated solutions are assigned a score that is equal to the product of the absolute value of the difference between the location-inventory cost of the dominated solution and the nondominated solution to which it is compared and the absolute value of the difference in uncovered demand volume between the two solutions. If one of the absolute values is zero (e.g., if the dominated solution has the same cost as the nondominated solution), the score is simply equal to the absolute value of the measure whose absolute value is nonzero (e.g., the uncovered demand in this case).

The genetic algorithm is based on five genetic operators: elite preservation, expansion, importation, crossover, and mutation. Each is described below. In addition, we applied an improvement algorithm to selected members of the population at each generation.

We implemented a generational replacement genetic algorithm, in which a new generation replaces the previous generation, as opposed to a steady-state algorithm in which the least desirable elements of the population are replaced as new solutions are identified. To initiate the construction of generation $t+1$, all nondominated solutions in generation t are copied to the emerging generation $t+1$. This is the *elite preservation operator*. Because the number of nondominated solutions is unknown a priori, there is a risk that a very large fraction of the solutions in generation $t+1$ will be nondominated, leaving little room for new solutions. To mitigate against this possibility, we implemented a new genetic algorithm operator that we term *expansion*. After the nondominated solutions from generation t are copied to generation $t+1$, we set the population size for generation $t+1$ to be the maximum of the population size at generation t and twice the number of nondominated (elite) solutions copied to generation $t+1$, as long as this maximum was less than 2,000. If the maximum exceeded

2,000, we capped the population size at 2,000. In the 31 runs summarized in §6 below, this cap was never reached, even though expansion occurred in 24 of the 32 instances.

The remainder of generation $t+1$ is filled using importation, crossover, and mutation. Twenty percent of the remaining solutions in generation $t+1$ are generated randomly using an *importation operator*. For each such solution, we first generate a random number that is used as a cutoff. Then, for each node, we draw a new random number. If the random number associated with a node is less than the cutoff value, we locate a facility at that node in the trial solution; otherwise, we do not. The purpose of the cutoff (which itself is uniformly distributed between zero and one as are all of the random numbers associated with each individual node) is to ensure that the distribution of the number of sites in the imported solutions is uniform. Each trial solution that is generated in this way is added to generation $t+1$, provided the solution is not already part of generation $t+1$.

The remaining solutions in generation $t+1$ are generated using crossover and mutation. Two different parents are selected at random from the generation t population with a bias toward the better solutions. The probability that solution k is selected as a parent from generation t is given by

$$\frac{2V_{\max}^t - V_k^t}{\sum_{k\in\mathbf{P}^t}(2V_{\max}^t - V_k^t)} = \frac{2V_{\max}^t - V_k^t}{2|\mathbf{P}^t|V_{\max}^t - \sum_{k\in\mathbf{P}^t} V_k^t},$$

where $V_{\max}^t$ is the largest value in the population at generation t, V_k^t is the value of solution k at generation t, and $\mathbf{P}^t$ is the population of solutions at generation t. We ensure that the two parents are not the same by rejecting the second parent if it is the same as the first and sampling again until a different parent is identified. A random *crossover* point is selected and the genes to the left of the crossover point of the first selected parent and those to the right of the crossover point of the second selected parent are merged to create a child solution. Then, with probability equal to the mutation probability, the solution is *mutated*. If the solution is to be mutated, a single gene is selected at random and the value of this gene is switched (from one to zero or vice versa). Child solutions generated in this manner are added to the population of generation $t+1$, provided the solution is not already part of generation $t+1$.

Once generation $t+1$ has been fully populated, with the exception of one set of runs specifically identified in the results section, we apply one or both of two *improvement* operators to a randomly selected member of the population that has not been improved before as well as to all supported solutions after an initial evaluation of generation $t+1$. (Note that improving one solution may result in another solution no longer being supported.) Recall that a supported solution is a nondominated solution that is also on the convex hull of the solutions when the solutions are plotted in objective space. (Note that elite solutions that are carried over from generation t might have been improved in an earlier generation.) The first improvement operator, which we term a *change*, considers changing every gene, one at a time, from its current state to the opposite state (from zero to one or vice versa) to see if any such change will reduce at least one of the two objectives—the location-inventory cost or the uncovered demand—without increasing the other objective. If such a change is found, it is immediately implemented and the search for additional improvements continues with the next gene. The search is repeated until no gene can be altered to improve at least one objective without degrading the other. If the improved solution differs from a solution in the population, it is saved as improved; otherwise, the unimproved solution remains in the population. In either case, the solution is identified as having been improved so we do not subsequently attempt to improve the solution. In addition to the randomly selected solution, all supported solutions that have not been subjected to the improvement operator are also improved in this way. For supported solutions, we also considered a second improvement operator that we call a *swap*, in which all possible ways of swapping a facility that is in the solution for one that is not in the solution are considered. As indicated above, in one set of runs, no improvements were attempted on any members of the population so that we could assess the impact of the pure genetic algorithm, without the improvement operators. In the runs in which improvement was attempted, we improved only one solution per generation, in addition to the supported solutions.

To generate each solution of the initial population, we first pick the number of facilities to be in the solution from a uniform distribution between one and the number of candidate locations. Then, we randomly select that number of candidate sites to be facilities. This process is repeated for each member of the population. For the initial population, all supported solutions are improved and an additional 50 solutions (out of an initial population of 100) are also improved using both improvement operators, changes, and swaps.

The genetic algorithm terminates when either of two conditions is satisfied. The algorithm is stopped if a user-specified number of generations have been generated. Alternatively, the algorithm is stopped if the number of consecutive generations in which no change in the set of nondominated solutions has been found exceeds a (different) user-specified value.

6. Computational Results

6.1. Input Conditions

Five realistically sized data sets were used to test the model outlined above. Basic characteristics of the data sets as well as key input parameters for the runs are shown in Table 1. Great circle distances were used in all five data sets. The data sets ranged in size from 49 nodes to 263 nodes. Sortcap is a 49-node data set representing the 48 capitals of the contiguous United States as well as Washington, D.C. The population at each node is equal to the 1990 population of the state (or city in the case of Washington, D.C.). The fixed facility location cost is equal to the 1990 median home value in the city. City1990 is the Sortcap data set expanded to 88 nodes to include the 50 largest cities in the continental United States according to the 1990 census. The population at each node in this data set equals the 1990 city population. 150City represents the 150 largest cities in the continental United States according to the 1990 census. Again, a node's population is equal to the city's population. All fixed costs, however, are set equal to $100,000 in 150City. Two different coverage distances were tested for this data set—150 miles and 300 miles. USA2000 263 Big Cities represents the 263 largest cities in the continental United States according to the 2000 census. Populations again correspond to the city population and all fixed costs again equal $100,000. Finally, Europe150 includes the 150 largest cities in Europe. As before, a node's population is equal the city population and all fixed costs equal $100,000. The coverage distance for this data set was set to 150 kilometers.

Table 2 gives the values of the model parameters that are common to all five data sets. In all cases, the *demand* at node i (μ_i) is set equal to three demands per day per million people in the population.

Table 3 summarizes the Lagrangian parameter settings for the computational results. These parameters can be used in the SITATION software (Daskin 2004).

6.2. Lagrangian Model Results

Table 4 contains a summary of the Lagrangian results for the five test problems for which the optimal trade-off curve was identified. After running for more than three days or more than 260,000 seconds, the program had not identified more than about half a dozen supported points on the optimal trade-off curve for the European data set. When we relaxed the optimality criterion for this data set, as outlined in §6.3, we identified over 30 noninferior solutions for this data set. The limited results shown in Table 4 for the European data set are based on the merging of the

Table 1 Summary of Test Data

Data file	Nodes	Total population	Total average daily demand	Distance metric	Coverage distance
Sortcap	49	247,051,601	741.15	miles	300
City1990	88	44,840,571	134.52	miles	300
150City (300 miles)	150	58,196,530	174.59	miles	300
150City (150 miles)	150	58,196,530	174.59	miles	150
Europe150	150	77,969,385	233.91	kilometers	150
USA2000 263 Big Cities	263	78,396,814	235.19	miles	300

Table 2 Model Parameters Common to All Runs

Parameter	Value
Demands per unit population per day	0.000003
Lead time (days)	5
Variance to mean ratio	1
z_α	1.96
Holding cost per unit per year	100
Fixed order cost at a DC	250
Dollars per demand mile in local delivery	0.01
Days per year	365
Fixed cost per shipment to a DC	10
Per unit cost of shipments to a DC	5

Table 3 Lagrangian Parameters Common to All Runs

Key Lagrangian parameters	
Substitution option at end of Lagrangian	Exchange algorithm
Critical percentage difference	0
Number of iterations	400
Number of iterations at root node	1,200
Minimum alpha value allowed	1.00E–08
Number of failures before changing alpha	12
Number of failures before changing alpha at root node	36
Crowder damping term	0.3
Restart failure count on improved solution	TRUE
Assignment search on improved solution	TRUE
Root node forcing	TRUE
CONSTANT in initial Lagrange multipliers	0
SLOPE in initial Lagrange multipliers	0
AVERAGE in initial Lagrange multipliers	10
DEMAND in initial Lagrange multipliers	0
FIXED in initial Lagrange multipliers	10
Do Lagrangian bounding	TRUE

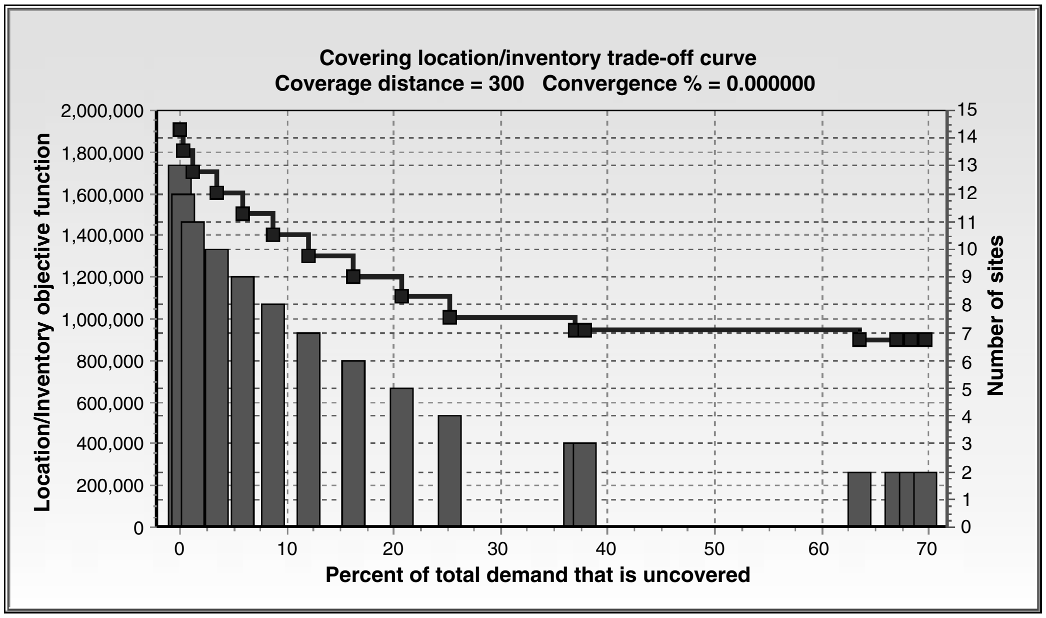
trade-off curves found in eight runs—two using the Lagrangian approach with relaxed optimality criteria as shown in Table 6 and six using the genetic algorithm with varying termination criteria as shown in Table 9. In the merged data set there were 52 solutions that were supported relative to the other solutions. The number of facilities in these 52 solutions ranged from 1 to 52. The time required to solve for the entire trade-off curve generally increases as the number of nodes increases. For the two data sets with under 100 nodes, the total solution time was under five minutes. For the larger data sets with 150 nodes and 263 nodes, the solution time ranged from more than 9 minutes to more than 32 hours. (All computation times are on a Dell Latitude C640 laptop computer running at 2.0 GHz. The program was coded in Delphi.) The number of Lagrangian iterations and the number of branch-and-bound nodes required to solve for the entire trade-off curve exhibit similar patterns. The computation time increases as the coverage distance decreases because additional solutions are typically found.

In addition to information about the computation times, Table 4 also provides information on the number of supported solutions found for each data set. For the 4 data sets based on U.S. data and a coverage distance of 300 miles, the number of solutions ranged from 12 to 19; when the coverage distance decreased to 150 miles for 150City, 39 supported solutions were identified. For the European data with a coverage distance of 150 kilometers or about 90 miles, 52 supported solutions were identified in the merged trade-off curve as described above. The minimum number of sites—the number associated with the minimum cost solution—ranged from 1 to 5 while the maximum number—the number of sites needed to cover all demands within the coverage distance—ranged from 13 to 52. As expected, the maximum number of sites increases as the coverage distance decreases because more facilities are needed to cover all demands. Note also that while many solutions involved assigning demand nodes to facilities other than the closest one, the aggregate percent of all assignments that were to nonclosest facilities was under 5% in all cases for which we could identify the optimal trade-off curve. Finally, we selected one solution from the 150City (300 mile) case in which 12 of the 150 nodes were assigned to facilities other than the closest facility. Forcing these 12 nodes to assign to the

Table 4 Summary Results for the Six Test Problems

Data file	Number of nodes	Solution time (sec.)	Lagrangian iterations	Branch-and-bound nodes	Number of supported solutions	Minimum number of sites	Maximum number of sites	Number of solutions with nonclosest assignments	Number of nonclosest assignments	Nonclosest as % of all assignments (%)
Sortcap	49	48.59	15,327	49	12	5	13	6	9	1.53
City1990	88	239.12	41,359	133	19	2	15	15	40	2.39
150City (300 miles)	150	569.15	43,124	131	16	2	13	12	61	2.54
150City (150 miles)	150	7,515.62	530,525	2,111	39	2	38	37	266	4.55
Europe150	150	N/A	N/A	N/A	52	1	52	N/A	N/A	1.62
USA2000 263 Big Cities	263	118,332.58	2,892,080	12,737	19	1	15	14	81	N/A

Figure 2 Results for 150City Data Set with Coverage Distance of 300 Miles



closest facility increased the cost from \$1,906,878 to \$1,908,747, a difference of \$1,869 or less than 0.1%. We believe that this justifies our choice of not implementing a search for nonclosest assignments in the genetic algorithm.

Figure 2 shows typical results that we obtained using the Lagrangian approach. The figure shows the trade-off curve for the 150City data set and a coverage distance of 300 miles. The step function is the location-inventory model objective function while the bar graph, with a scale on the right-hand side of the figure, is the number of facilities located. The minimum cost solution is at the far right of the figure; two facilities are located at a total annual cost of approximately \$896,000 leaving more than 44,385 annual demands (or nearly 70% of the total annual demand) uncovered. This solution is shown in Figure 3. However, for less than a 6% increase in annual cost, the uncovered demand volume can be reduced to less than 38% of the total annual demand. This entails locating one additional facility, as shown in Figure 4. On the other hand, covering all demands necessitates a 113% increase in cost and the siting of 13 facilities.

Figure 5 shows the trade-off curve that resulted from using the 150City data set with a coverage distance of 150 miles. Again, significant reductions in the uncovered demand can be achieved at relatively little increase in cost, while covering all demands requires a very large cost increase. With the smaller coverage distance, the increase in cost associated with covering all demands is even greater (more than 400% more than the minimal cost).

6.3. Effects of Upper Bounding and Different Optimality Gaps on the Lagrangian Solutions

In this section, we evaluate the impact of changing the Lagrangian convergence criterion and using or not using the bounding rule outlined in §4, which states that the upper bound on any solution that lies between two solutions (U^1, C^1) and (U^2, C^2) is $C^1 + WU^1 = C^2 + WU^2$. The effects are evaluated in terms of three measures of the solution difficulty—the solution time, the number of branch-and-bound nodes evaluated, and the number of Lagrangian iterations needed to identify the trade-off curve—and one measure of the solution quality, the number of solutions found. Table 5 presents the results of tests applied to the 150City data set with a service distance of 300 miles. The solution time, number of branch-and-bound nodes, and number of Lagrangian iterations all go down as the convergence criterion is relaxed from optimality (0.0%) to 0.5% and, finally to 1.0%. The three measures all increase by about 50% if

Figure 3 **Minimal Cost Solution for 150City Data Set with a Coverage Distance of 300 Miles**

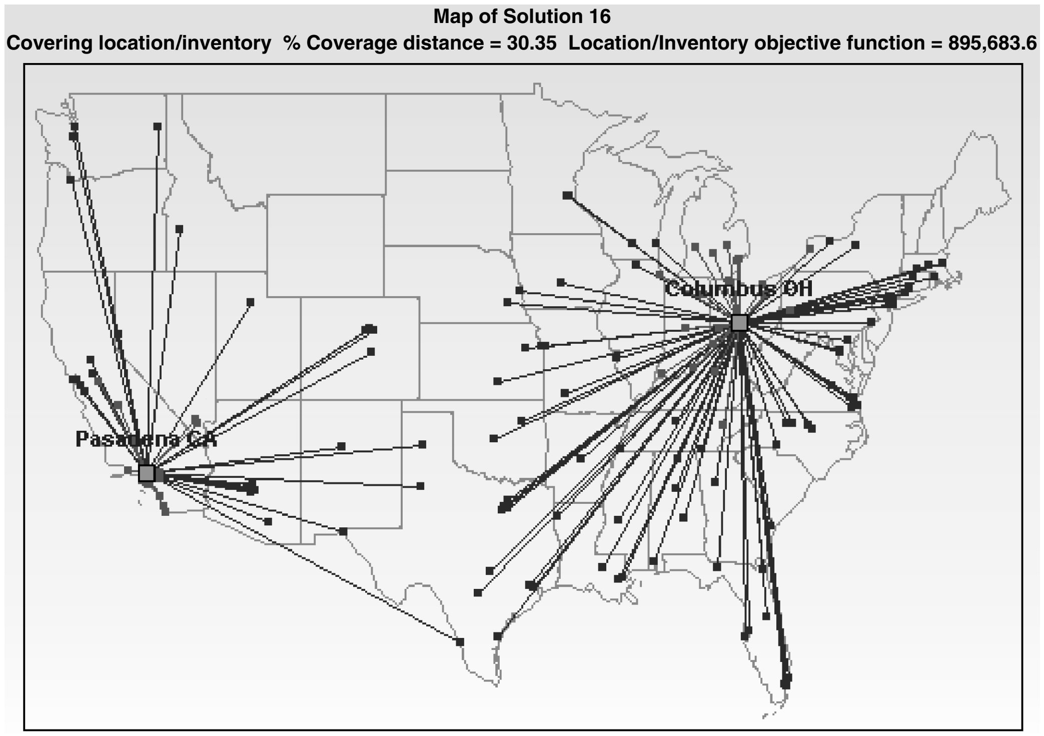

Map of Solution 16
Covering location/inventory % Coverage distance = 30.35 Location/Inventory objective function = 895,683.6
Columbus OH
Pasadena CA

Figure 4 **A Solution that Reduces Uncovered Demand by 47%, While Increasing Cost by Less Than 6%**

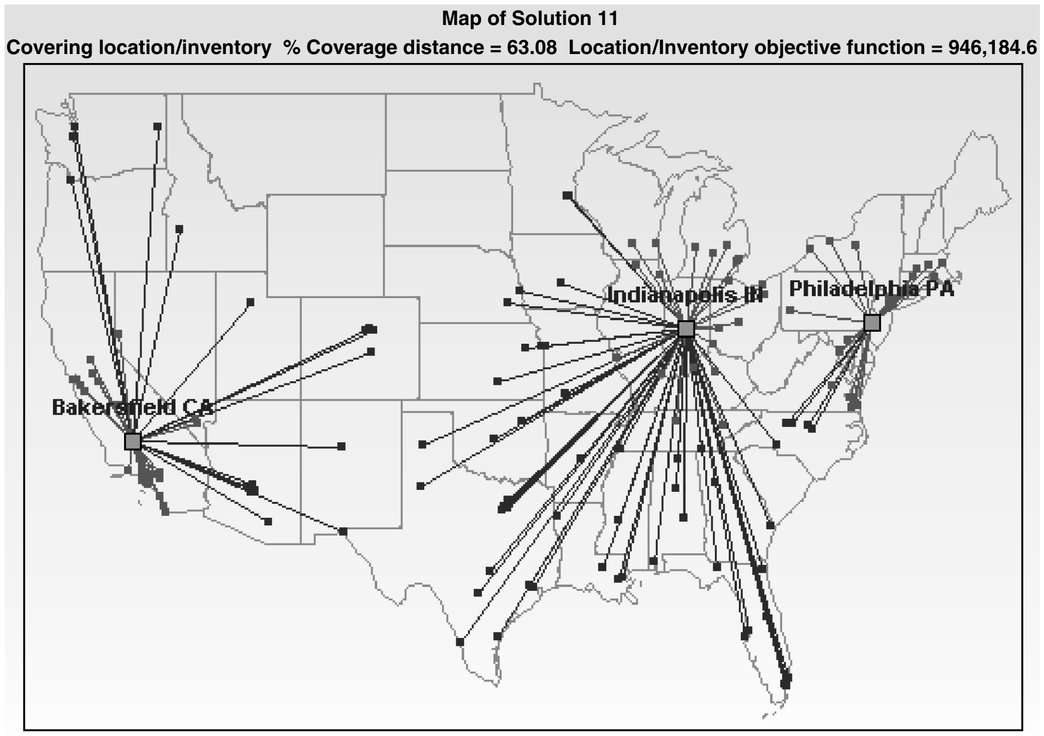
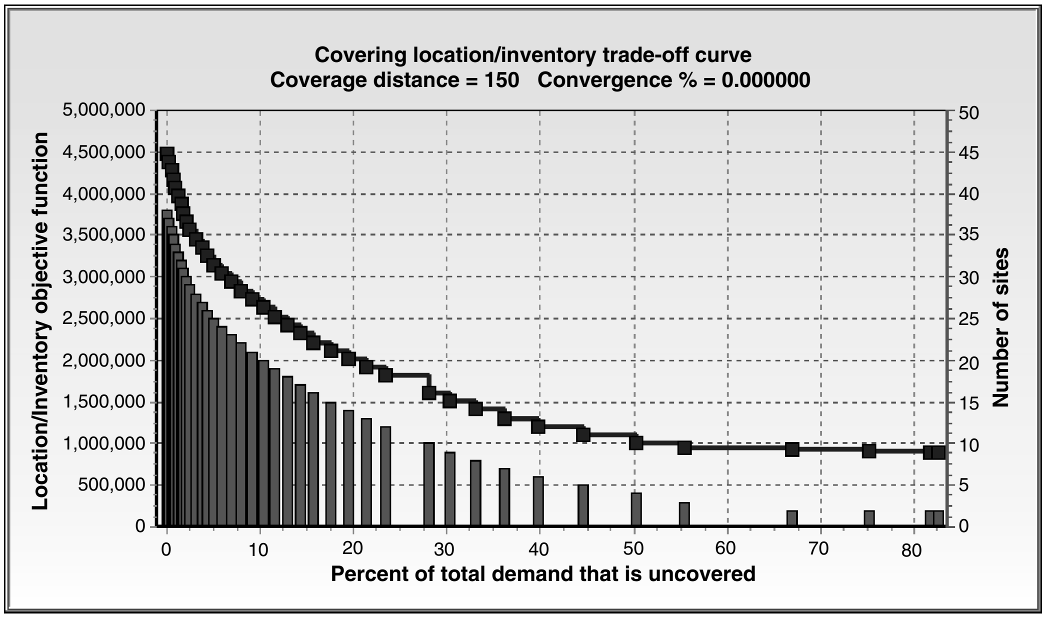

Map of Solution 11
Covering location/inventory % Coverage distance = 63.08 Location/Inventory objective function = 946,184.6
Indianapolis IN
Philadelphia PA
Bakersfield CA

Figure 5 Results for 150City Data Set with Coverage Distance of 150 Miles



bounding is not employed. Bounding has no effect on the final trade-off curve that is identified if the optimality gap is zero; however, for nonzero optimality gaps, the number of solutions found on the trade-off curve depends on the optimality gap and on whether or not bounding is used. For nonzero optimality gaps, the trade-off curve that is found is an approximation of the actual curve. At least in the limited testing reported below, the number of solutions found for these approximate curves is less than the number identified in the optimal curve.

Despite the fact that the trade-off curve identified with an optimality gap of zero is the "optimal" curve, it does not necessarily include all noninferior solutions, as there may be solutions in the duality gap region. This is region *B* of Figure 1. In the experiments outlined below, one such solution was identified. When the optimality gap was set to 1% and bounding was not used, one additional noninferior point was identified. It lies between the fourth and fifth least costly solutions shown in Figure 2, which depicts the optimal trade-off curve for the 150City data set with a coverage distance of 300 miles. Thus, while allowing a nonzero optimality gap will result in a trade-off curve that is not guaranteed to be optimal, doing so may allow the analyst to find solutions, which would not otherwise be found using the weighting method.

To test the effect of the optimality gap further, we varied the optimality gap in all five test problems for which we could find the optimal solution using Lagrangian relaxation as well as the results for the Europe150 data set for which an exact solution is not known. The results are shown in Table 6. The first three columns describe the problem. The fourth column gives the optimality gap. The solution time, number of Lagrangian iterations, and number of branch-and-bound nodes are shown in Columns 5 through 7.

Table 5 Effects of Different Optimality Gaps and Bounding Using 150City Data Set with a Coverage Distance of 300 Miles

Solution times (sec.)			B&B nodes		
Optimality gap (%)	Bounding	No bounding	Optimality gap (%)	Bounding	No bounding
0.00	569.15	912.36	0.00	131	213
0.50	135.54	212.71	0.50	25	36
1.00	79.50	109.33	1.00	15	20
Iterations			Number of solutions found		
0.00	43,124	62,711	0.00	16	16
0.50	10,244	14,578	0.50	11	12
1.00	6,094	7,456	1.00	8	10

Table 6 Tests of Optimality Gap on Solution Time and Solution Quality

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Data set	Number of nodes	Coverage distance	Percent optimality	Solution time (sec.)	Lagrangian iterations	Branch-and-bound nodes	Number of solutions	Number of solutions supported w.r.t. set found	Number of optimal supported solutions found	Number of solutions with same coverage	Percent loc/inv error for those with same coverage (%)	Max % loc/inv error for those with same coverage (%)	Number of solutions with nonclosest assignment	Number of demand nodes with nonclosest assignment
Sortcap	49	300	1	12.69	3,643	17	9	8	7	8	0.05589	0.44716	6	9
			0.5	16.14	4,756	21	11	10	9	10	0.04472	0.44716	6	11
			0.1	18.00	5,376	23	12	12	12	12	0.00000	0.00000	6	9
			0.01	19.21	5,698	23	12	12	12	12	0.00000	0.00000	6	9
			Optimal	48.59	15,327	49	12	12	12	N/A	N/A	N/A	6	9
City1990	88	300	1	82.59	14,187	43	14	14	14	14	0.00000	0.00000	10	27
			0.5	114.01	19,578	65	16	16	15	16	0.00317	0.05066	12	33
			0.1	173.79	30,005	99	19	18	18	18	0.00000	0.00000	15	40
			0.01	184.56	31,942	107	19	19	19	19	0.00000	0.00000	15	40
			Optimal	239.12	41,359	133	19	19	19	N/A	N/A	N/A	15	40
150City	150	300	1	79.50	6,094	15	8	8	4	8	0.02974	0.18387	6	37
			0.5	135.54	10,244	25	11	11	7	11	0.02163	0.18387	9	50
			0.1	218.37	16,569	51	15	15	14	15	0.00091	0.01368	11	60
			0.01	306.65	23,307	71	16	16	15	16	0.00005	0.00075	12	61
			Optimal	569.15	43,124	131	16	16	16	N/A	N/A	N/A	12	61
150City	150	150	1	247.22	18,183	45	23	19	14	4	0.02087	0.07355	21	147
			0.5	220.18	15,127	37	19	19	3	19	0.02963	0.25756	18	116
			0.1	418.18	29,201	61	31	31	11	31	0.00560	0.04995	30	215
			0.01	3,632.81	245,199	883	39	39	23	39	0.00089	0.00891	37	272
			Optimal	7,515.62	530,525	2,111	39	39	39	N/A	N/A	N/A	37	266
Europe150	150	150 km	1	875.96	58,583	123	33	30	15	24	0.00434	0.01817	28	249
			0.5	4,167.46	241,724	816	34	34	24	31	0.00152	0.01254	29	218
			Best known				52	52	52				N/A	N/A
USA2000 263 Big Cities	263	300	1	7,973.41	177,041	624	17	16	8	12	0.01224	0.11109	13	93
			0.5	20,173.59	475,573	1,817	20	17	8	14	0.01305	0.09880	16	114
			0.1	65,816.99	1,628,549	6,671	20	19	13	19	0.00438	0.02741	15	105
			0.01	107,642.95	2,612,143	11,275	19	19	17	19	0.00010	0.00112	14	78
			Optimal	118,332.58	2,892,080	12,737	19	19	19	N/A	N/A	N/A	14	81

The remaining eight columns give an indication of the quality of the solution curve. Column 8 gives the number of supported solutions found by the algorithm with the indicated optimality gap. The final row for each data set gives the optimal set of supported solutions (or the best-known solution in the case of the Europe150 data set). When the optimality gap is greater than zero, the algorithm may find solutions that are not supported even with respect to the other solutions found. In other words, the algorithm can find a solution that ends up in region B of Figure 1. Column 9 gives the number of solutions that are supported with respect to the other solutions found in that run. Not all of the supported solutions reported in Column 8 are optimal. This number is reported in Column 10.

The next three columns provide a measure of the deviation of the suboptimal trade-off curve from the optimal set of supported solutions. Column 11 reports the number of solutions found that had the same total demand coverage as one of the optimal supported solutions. For example, for the 49-node Sortcap data set, with an optimality gap of 1%, the algorithm found 8 solutions that had the same total coverage as one of the 12 optimal supported solutions. The next two columns report the average and maximum percentage error in the location-inventory cost objective for the suboptimal solutions whose coverage corresponds to that of an optimal supported solution.

Finally, Column 14 reports the number of solutions (out of the total number of solutions found reported in Column 8) that entailed assigning any nodes to a nonclosest facility. Column 15 reports the total number of such nonclosest assignments over all solution assignments (which equals the number of solutions times the number of nodes given in Column 2). The percentage of all assignments that are to nonclosest facilities is under 5%, except for the Europe150 solution run with a 1% convergence criterion for which the percentage is 5.03%. Averaged over all runs, 3.14% of the assignments are to nonclosest facilities.

While the quality of the solution degrades as the optimality gap increases in all cases, the solution quality remains high for optimality gaps below about 0.5%. Columns 11, 12, and 13 of the table indicate that most of the suboptimal solutions have the same uncovered demand as one of the optimal solutions and that the location-inventory cost differs from the optimal cost by under 0.06% on average and under 0.5% in all cases reported. However, increasing the optimality gap significantly reduces the solution time. Increasing the optimality gap from 0.0% to 0.01% reduced the solution time more than 30% on average. At an optimality gap of 0.5%, the solution time was approximately 75% less than the time required for the optimal solution. These results suggest that very good approximations to the trade-off curve can be found by reducing the optimality gap in the Lagrangian algorithm.

6.4. Genetic Algorithm Results

The genetic algorithm was tested on each of the six test problems identified in Table 1. The parameters for the genetic algorithm are shown in Table 7. All runs, except one, were done using the improvement operators described in §5. To test the impact of the genetic algorithm without improvements, we ran a series of runs on the largest problem, the U.S. 263-node data set.

Table 7 Basic Genetic Algorithm Parameters

Parameter	Parameter value
Initial population size	100
Mutation probability	0.1
Fraction of nonelite solutions generated via importation	0.2

Table 8 Termination Criteria for Genetic Algorithm Cases

Case	Maximum number of generations	Maximum number of generations without a change in the set of nondominated solutions
1	100	100
2	1,000	100
3	2,500	100
4	5,000	100
5	10,000	100
6	100,000	1,000

To test the effect of running the genetic algorithm for more iterations, a number of different termination criteria were considered, as shown in Table 8.

Table 9 presents the results of our experiments with the genetic algorithm. For the U.S.-based data sets, m rows of results are given. The first $m-2$ rows correspond to cases in which the genetic algorithm terminated when either the maximum allowable number of generations was reached or 100 consecutive generations passed without any change in the trade-off curve. The $m-1$ row corresponds to Case 6 of Table 8 in which we set a large limit on the maximum number of generations (100,000) and terminated the genetic algorithm when 1,000 consecutive generations had passed without a change in the trade-off curve. The final row summarizes key characteristics of the optimal solution found using the Lagrangian procedure outlined above. Recall that the Lagrangian procedure finds all supported solutions on the trade-off curve, but does not find any nonsupported but nondominated solutions. For the European data set, the first four rows correspond to Cases 1 through 4 outlined in Table 8. Row 5 corresponds to Case 6. For Row 6, we allowed the genetic algorithm to generate 100,000 generations. Finally, Row 7 summarizes the best-known results. The last block of results are those for the large U.S. data set without the use of any of the improvement operators.

The columns in Table 9 are, for the most part, similar to those in Table 6. Columns 4, 5, and 6 summarize the nominal number of generations, the actual

Table 9 Genetic Algorithm Results

1	2	3	4	5	6	7	8	9	10	11	12	13	14
Data set	Number of nodes	Coverage distance	Nominal number of generations	Actual number of generations	Reason for termination	Solution time (sec.)	Final population size	Number of nondominated solutions	Number of supported solutions w.r.t. set found	Number of optimal supported solutions found	Number of solutions with same coverage	Percent loc/inv error for those with same coverage (%)	Max % loc/inv error for those with same coverage (%)
Sortcap	49	300	100	100	Gen	5.49	100	35	8	4	8	0.199	0.996
			1,000	422	100 No Imp	18.93	100	40	10	4	10	0.086	0.227
			100,000	2,710	1,000 No Imp	113.16	100	46	11	5	11	0.078	0.227
			Optimal			48.59			12				
City1990	88	300	100	100	Gen	8.98	100	39	11	2	6	1.680	5.049
			1,000	987	100 No Imp	49.56	158	79	17	2	9	1.189	4.033
			100,000	13,453	1,000 No Imp	659.88	200	100	20	4	19	0.118	1.537
			Optimal			239.12			19				
150City	150	300	100	100	Gen	17.17	100	30	9	0	2	5.274	10.544
			1, 000	666	100 No Imp	47.80	148	74	15	1	6	1.925	6.105
			100,000	10,876	1,000 No Imp	641.04	232	116	15	3	14	0.022	0.084
			Optimal			569.15			16				
150City	150	150	100	100	Gen	28.73	154	74	17	1	4	4.162	6.924
			1,000	1,000	Gen	92.07	362	180	28	2	15	0.334	4.688
			2,500	2,500	Gen	199.35	430	212	31	2	25	0.028	0.082
			5,000	3,866	100 No Imp	298.48	448	222	33	2	30	0.031	0.082
			100,000	34,624	1,000 No Imp	2,955.38	694	347	37	2	35	0.033	0.082
			Optimal			7,515.62			39				
Europe150	150	150 km	100	100	Gen	33.54	182	89	20	1	3	1.264	3.787
			1,000	1,000	Gen	113.24	658	329	37	6	19	0.323	3.142
			2,500	2,500	Gen	272.36	816	402	41	6	17	0.003	0.008
			5,000	3,596	100 No Imp	392.23	830	415	41	7	17	0.003	0.008
			100,000	19,662	1,000 No Imp	2,576.82	1,058	525	51	13	25	0.095	2.318
			100,000	100,000	Gen	14,481.30	1,114	557	51	30	51	0.002	0.008
			Best known			N/A			52				
USA2000 263 Big Cities	263	300	100	100	Gen	60.05	100	41	12	1	4	11.977	28.960
			1,000	1,000	Gen	137.93	172	86	17	4	6	3.158	18.945
			2,500	2,500	Gen	257.40	252	124	19	4	6	1.053	6.314
			5,000	2,660	100 No Imp	270.61	252	123	19	4	6	1.053	6.314
			100,000	21,113	1,000 No Imp	2,199.37	438	219	18	4	14	0.453	6.314
			Optimal			118,332.58			19				
USA2000 263 Big Cities No improvement	263	300	100	100	Gen	4.24	100	34	14	0	2	62.283	63.550
			1,000	1,000	Gen	43.39	276	139	17	3	4	12.705	50.820
			2,500	2,500	Gen	130.08	344	134	20	4	6	6.346	38.074
			5,000	5,000	Gen	283.37	344	143	22	5	9	3.524	31.710
			10,000	7,712	100 No Imp	451.61	412	206	19	5	13	5.215	25.463
			100,000	22,439	1,000 No Imp	1,511.09	490	244	21	5	11	1.154	12.683
			Optimal			118,332.58			19				

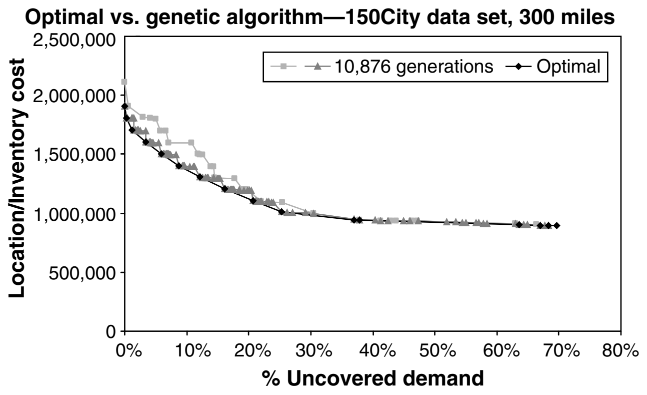
number of generations and the reason the algorithm terminated. Column 8 gives the final population size while Column 9 provides the number of nondominated solutions found. The final population size will typically be about twice the number of nondominated solutions provided the latter number exceeds 50. Column 12 reports the number of solutions found by the genetic algorithm that had the same total coverage as one of the optimal supported solutions. For example, for the 49-node Sortcap data set, the genetic algorithm found 4 solutions after 100 generations that had the same total coverage as one of the 12 optimal supported solutions. Columns 10, 11, 13, and 14 of Table 9 are similar to Columns 9, 10, 12, and 13 of Table 6.

In many cases, the number of nondominated solutions found by the genetic algorithm greatly exceeds the optimal number of supported solutions found by the Lagrangian procedure. For example, for the 150City data set with a coverage distance of 150 miles, when the genetic algorithm terminated after only 100 generations, it had found 74 nondominated solutions compared to 39 supported solutions in the optimal trade-off curve. After 3,866 generations (which took about five minutes of time), the genetic algorithm had found 222 nondominated solutions. The Lagrangian algorithm for this problem took more than two hours to find 39 optimal supported solutions.

To illustrate the quality of the genetic algorithm solutions further, Figure 6 compares the genetic algorithm trade-off curve to the optimal solutions for the 150City data set with a coverage distance of 300 miles after 100 and 10,876 generations. While the solution after 100 generations differs significantly from the optimal curve on visual inspection, the solution after 10,876 generations does not appear to differ appreciably from the optimal curve except for the fact that the curve from the genetic algorithm includes many nonsupported nondominated solutions.

Finally, Figure 7 shows the progress of the algorithm for this problem over the first 10,876 generations. Values shown are for every twentieth generation. The graph plots the number of consecutive generations with the same trade-off curve on the left-hand axis. On four occasions, beginning around generation 4,700, this exceeded 800 before attaining the termination criterion of 1,000 at generation 10,876. The right-hand axis plots the population size and the number of nondominated solutions. Both values increase rapidly through the first 3,000 generations and then begin to stabilize.

Figure 6 **Optimal and Heuristic Trade-off Curves for the 150City Problem with a Coverage Distance of 300 Miles**



7. Managerial Insights and Conclusions

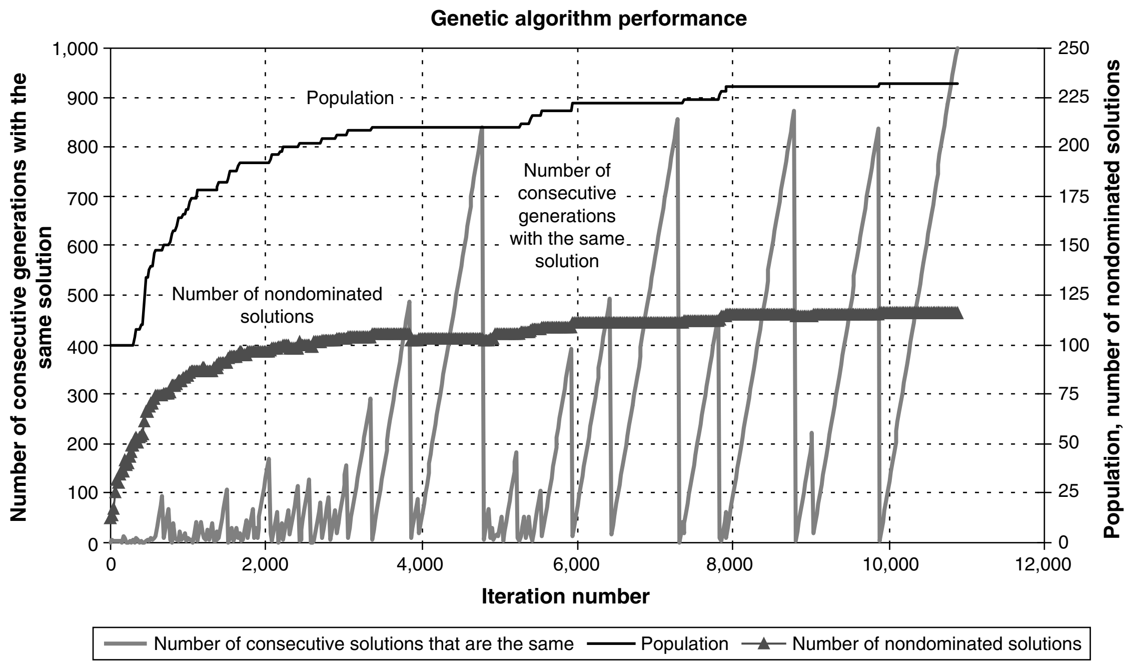
In this paper, we have extended a nonlinear integrated location-inventory model to incorporate a measure of customer service quality. Key costs represented by the model include fixed DC location costs, working and safety stock costs at the DCs, fixed and variable shipment costs from the plant to the DCs, and transportation costs from the DCs to the customers. Customer service is measured by the fraction of all customer demands that are within a specified distance or service standard of the DC to which they are assigned.

Through tests of the model on data sets ranging in size up to 263 demand nodes, we showed that it is important for a company to find the right trade-off between supply chain cost and service. The cost difference between the cost-minimization solution and the service-maximization solution can be quite large. In our tests, the service-maximization solution ranged from 6 to 20 times bigger than the cost-minimization solution. We also showed that significant improvements in customer service can often be achieved at relatively little cost. While this often entails locating additional DCs, the cost of the incremental facilities is largely offset by reduced outbound transportation costs.

Furthermore, we found that it can be very time consuming to find the optimal cost-service trade-off curve for large problem instances. In this case, our proposed genetic algorithm can be a very good alternative to solve the problem because it can generate high-quality solutions quickly. Furthermore, the number of nondominated solutions found by the genetic algorithm greatly exceeds the optimal number of supported solutions found by the Lagrangian procedure. We believe that similar algorithms can be developed for more general supply chain design problems.

As indicated earlier, many firms maintain different service levels for different types of customers (e.g.,

Figure 7 Progress of the Genetic Algorithm Over 10,876 Generations



2-hour service, 8-hour service, and 24-hour service). We are exploring ways of extending the model to examine the trade-off between cost and service for several different classes of customers.

References

Badri, M. A., A. K. Mortagy, C. A. Alsayed. 1998. A multi-objective model for locating fire stations. *Eur. J. Oper. Res.* **110** 243–260.

Belardo, S., J. Harrald, W. A. Wallace, J. Ward. 1984. A partial covering approach to siting response resources for major maritime oil spills. *Management Sci.* **30** 1184–1196.

Church, R. L., C. S. ReVelle. 1974. The maximal covering location problem. *Papers Regional Sci. Assoc.* **32** 101–118.

Cohon, J. L. 1978. *Multiobjective Programming and Planning*. Academic Press, New York.

Current, J., H. Min, D. Schilling. 1990. Multiobjective analysis of facility location decisions. *Eur. J. Oper. Res.* **49** 295–307.

Daskin, M. S. 1995. *Network and Discrete Location: Models, Algorithms, and Applications*. John Wiley and Sons, New York.

Daskin, M. S. 2004. SITATION—Facility location software. Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL. Available at http:// users.iems.nwu.edu/~msdaskin/.

Daskin, M. S., E. Stern. 1981. A hierarchical objective set covering model for EMS vehicle deployment. *Transportation Sci.* **15** 137–152.

Daskin, M., C. Coullard, Z. J. Shen. 2002. An inventory-location model: Formulation, solution algorithm and computational results. *Ann. Oper. Res.* **110** 83–106.

Fernandez, E., J. Puerto. 2003. Multiobjective solution of the uncapacitated plant location problem. *Eur. J. Oper. Res.* **145** 509–529.

Heller, H., J. L. Cohon, C. S. Revelle. 1989. The use of simulation in validating the multiobjective EMS location model. *Ann. Oper. Res.* **18** 303–322.

Hopp, W., M. L. Spearman. 1996. *Factory Physics: Foundations of Manufacturing Management*. Irwin, Chicago, IL.

Hultz, J. W., D. D. Klingman, G. T. Ross, R. M. Soland. 1981. An interactive computer system for multicriteria facility location. *Comput. Oper. Res.* **8** 249–261.

Jayaraman, V. 1999. A multi-objective model for a capacitated service facility logistics problem. *Internat. J. Physical Distribution Logist. Management* **29**(1) 65–81.

Lee, S. M., G. I. Green, C. Kim. 1981. A multiple criteria model for the location-allocation problem. *Comput. Oper. Res.* **8** 1–8.

Ma, Y.-H., G. R. Wilson. 2002. Neighborhoods: A new service parts stock plan. *INFORMS Annual Meeting*, San Jose, CA.

Nahmias, S. 1997. *Production and Operations Management*, 3rd ed. Irwin, Chicago, IL.

Nozick, L., M. Turnquist. 2001. Inventory, transportation, service quality and the location of distribtuion centers. *Eur. J. Oper. Res.* **129** 362–371.

ReVelle, C. S., J. L. Cohon, D. Shobrys. 1981. Multiple objectives in facility location: A review. *Organisations: Multiple Agents with Multiple Criteria. Lecture Notes in Economic and Mathematical Systems*, Vol. 190. Springer-Verlag, Berlin, Germany.

Ross, G. T., R. M. Soland. 1980. A multicriteria approach to location of public facilities. *Eur. J. Oper. Res.* **4** 307–321.

Sabri, E. H., B. M. Beamon. 2000. A multi-objective approach to simultaneous strategic and operational planning in supply chain design. *OMEGA* **28** 581–598.

Schilling, D. A. 1980. Dynamic location modeling for public-sector facilities: A multicriteria approach. *Decision Sci.* **11**(4) 714–724.

Shen, Z. J. 2000. Efficient algorithms for various supply chain problems. Ph.D. dissertation, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

Shen, Z. J., C. Coullard, M. Daskin. 2003. A joint location-inventory model. *Transportation Sci.* **37** 40–55.

Shu, J., C. P. Teo, Z. J. Shen. 2005. Stochastic transportation-inventory network design problem. *Oper. Res.* **53** 48–60.

Solanki, R. 1991. Generating the noninferior set in mixed biobjective linear programs: An application to a location problem. *Comput. Oper. Res.* **18** 1–15.

Stern, G. 1995. GM expands its experiment to improve Cadillac's distribution, cut inefficiency. *Wall Street J.* (February 8).