NORTHWESTERN UNIVERSITY

Supply Chain Robustness and Reliability: Models and Algorithms

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Industrial Engineering and Management Sciences

By

Lawrence V. Snyder

EVANSTON, ILLINOIS

December 2003

ABSTRACT

Supply Chain Robustness and Reliability: Models and Algorithms

Lawrence V. Snyder

Supply chain design models have traditionally treated the world as if we knew everything about it with certainty. In reality, however, parameter estimates may be inaccurate due to poor forecasts, measurement errors, changing demand patterns, or other factors. Moreover, even if all of the parameters of the supply chain are known with certainty, the system may face disruptions from time to time, for example, due to inclement weather, labor actions, or sabotage. This dissertation studies models for designing supply chains that are *robust* (i.e., perform well with respect to uncertainties in the data, such as demand) and *reliable* (i.e., perform well when parts of the system fail).

The first half of this dissertation is concerned with models for robust supply chain design. The first of these models minimizes the expected systemwide cost, including costs for facility location, transportation, and inventory. The second model adds a constraint that restricts the regret in any scenario to be within a pre-specified limit. Both models are solved using Lagrangian relaxation. The second model presents an additional challenge since feasible solutions cannot always be found easily, and it may even be difficult to determine whether a given problem is feasible. We present strategies for overcoming these difficulties. We also discuss regret-constrained versions of two classical facility location problems and suggest algorithms for these problems based on variable-splitting. The algorithms presented here can be used (heuristically) to solve minimax-regret versions of the corresponding problems.

In the second half of the dissertation, we present a new approach to supply chain optimization that attempts to choose facility locations so that if a distribution center becomes unavailable, the resulting cost of operating the system (called the "failure cost") is not excessive. We discuss two types of reliability models, one that considers the maximum failure cost and one that considers the expected failure cost. We propose several formulations of the maximum failure cost problem and discuss relaxations for them. We also present a tabu search heuristic for solving these problems. The expected failure cost problem is solved using Lagrangian relaxation. Computational results from both models demonstrate empirically that large improvements in reliability are often possible with small increases in cost.

Dissertation Committee

Professor Mark S. Daskin, Committee Chair
Department of Industrial Engineering and Management Sciences
Robert R. McCormick School of Engineering
Northwestern University

Professor Collette Coullard
Department of Industrial Engineering and Management Sciences
Robert R. McCormick School of Engineering
Northwestern University

Professor Karen Smilowitz
Department of Industrial Engineering and Management Sciences
Robert R. McCormick School of Engineering
Northwestern University

Professor Chung–Piaw Teo
Department of Decision Sciences
National University of Singapore

Acknowledgments

I would like to thank Mark Daskin, without whose guidance this project would have been impossible. Mark has been generous and supportive, academically, professionally, and personally, and is a true role model to me.

My readers, Collette Coullard, Karen Smilowitz, and Chung-Piaw Teo, have been of great help throughout my graduate career. I especially appreciate the time that Collette took to mentor me as a first-year graduate student. Karen's perspective on my research has helped my dissertation take shape as a cohesive whole, and her support got me through the job search process. C.P. has been a valuable resource whenever I ran into a roadblock, and I value his impressive expertise.

I would like to thank my parents, Harvey and Lenny, my sister Tanya, and my extended family, Carol, Art, Joyce, and Amy. This achievement is due in no small part to their love, support, and generosity.

I would not have stayed happy and sane throughout this process without our friends Mark and Alyssa, for whom a deck of cards and a bottle of Rancho will always be waiting at our home. Our friend and choir conductor, Randi, has been a musical and personal inspiration, and I will miss him.

My advisors during my undergraduate years at Amherst College, Norton Starr and Ruth Haas, were formative in my academic development, and I thank them for helping me develop as a student and researcher, and for encouraging me to pursue graduate study in Operations Research. I also wish to thank the rest of the faculty and my fellow graduate students in the IE/MS department for creating a productive environment of intellectual growth and for supporting me as a student, teacher, and researcher.

Most especially, I want to thank my wife, Suzanne, an amazing friend and partner from whom I have learned so much about generosity, dedication, and scholarship. I hope I can repay some of the support, encouragement, and love she has given me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Supply chain network design decisions are costly and difficult to reverse, and their impact spans a long time horizon. During the time when design decisions are in effect, any of the parameters of the problem—costs, demands, distances, lead times—may fluctuate widely. However, supply chain optimization models have traditionally treated the world as if we knew everything about it with certainty. Even inventory models, the main aspect of supply chain management whose inherent uncertainty has been considered widely in the literature, generally assume that probability distributions of the uncertain parameters are known. In reality, however, parameter estimates may be inaccurate due to poor forecasts, measurement errors, changing demand patterns, or other factors. Moreover, even if all of the parameters of the supply chain are known with certainty, the system may face disruptions from time to time, for example, due to inclement weather, labor actions, or sabotage. This highlights a need for models that incorporate various forms of uncertainty into strategic decisions about supply chain design. In this dissertation, we present models for robust and reliable supply chain design. Broadly defined, a supply chain is *robust* if it performs well with respect to uncertain future conditions—for example, costs or travel times; a supply chain is *reliable* if it performs well when parts of the system fail—for example, when a distribution center becomes unavailable because of poor weather.

The goal of *robust* optimization in general is to find solutions that perform well under every realization of the uncertain parameters, though not necessarily optimally in any. Our models for robust supply chain design use as an underlying model a joint location–inventory model that recently appeared in the literature; this model, called the Location Model with Risk Pooling (LMRP), chooses distribution center (DC) locations, assignments of retailers to DCs, and inventory levels at the DCs to minimize the total cost. The first model we consider in this dissertation solves the LMRP in a stochastic context to minimize the total expected systemwide cost. The second model is similar to the first but it also imposes a maximum level of "regret," or opportunity loss, on each scenario. Both of these are two-stage models, in that strategic decisions (facility location) must be made now, before it is known which scenario will come to pass, while tactical decisions (assignment of retailers to DCs, setting inventory levels) are made in the future, after the uncertainty has been resolved. However, by simply multiplying the results by the number of time periods in the planning horizon, one can think of these as multi-period models in which we make strategic decisions now and then make separate tactical decisions in each time period. Thus our models do not require parameters to be time-stationary; demand means and variances may change over time. If we make decisions now that hedge poorly against the various scenarios, we must live with the consequences as parameters change.

The notion of supply chain *reliability* has not previously been studied. While several recent authors have identified a need for strategies that reduce the vulnerability of supply chains to disruptions, few quantitative models have been developed. Our research begins to address this need by formulating several models for choosing facility locations so that if one or more facilities becomes unavailable, the remaining system is adequate to serve customers efficiently. Naturally, systems designed based solely on cost may not be reliable, and more reliable systems may be more expensive to operate. There is a tradeoff between the day-to-day operating cost of a system and the reliability of the system. One set of models in this dissertation addresses the tradeoff between operating cost and the *maximum* cost that might result when a facility fails, while another set addresses that between operating cost and the *expected* cost of failure when facilities have a given probability of failing. All of these models have a classical facility location model as the underlying model. Future research will extend the reliability concept to richer supply chain models such as the LMRP described above.

1.1 Robust vs. Reliable Optimization

As noted above, *robust* optimization is concerned with finding solutions that perform well with respect to uncertain future conditions, while *reliable* optimization is concerned with finding solutions that perform well when parts of the system fail. Typically, robust optimization problems involve scenarios that describe the uncertain parameters; these scenarios may be discrete (each scenario providing a complete description of the parameters, possibly with a probability of each scenario's occurrence) or continuous (each parameter represented by a range of values that it may take, possibly described by a probability distribution). While some authors use the term "stochastic" for problems involving expected performance and reserve the term "robust" for problems involving worst-case performance or regret, we prefer to use "robust" more broadly to refer to any solution that performs well across a range of scenarios, in expected performance, worst-case performance, or any of a number of other measures that have appeared in the literature over the past half-century or so. In contrast, "reliability" refers to a different approach to uncertainty in which we are hedging against failures in the system described by a given solution. In that sense, one can view robustness as concerned with uncertainty in the data, while reliability refers to uncertainty in the solution itself. Another way to view the distinction in the context of supply chain design or facility location is that robustness is concerned with demand-side uncertainty (uncertainty in demands, costs, or other aspects of the distribution of goods from a firm to its customers) while reliability is concerned with supply-side uncertainty (uncertainty in the availability of plants, distribution centers, or other facilities required to produce and distribute the product).

This discussion is meant to provide a framework for thinking about the distinction between robustness and reliability. It is not meant to suggest that there is a rigorous distinction between the two from a modeling or optimization point of view. Indeed, one could incorporate facility failures into the problem data, blurring the "data uncertainty vs. solution uncertainty" distinction above. Similarly, the line between supply-side and demand-side may itself be blurred, making that distinction irrelevant, as well. Still, we find these to be useful distinctions between the approaches to uncertainty, and we encourage the reader to keep them in mind while reading this dissertation.

1.2 Research Contributions

Supply chain design models mainly consider deterministic settings, despite the long horizons over which supply chains are operational and the strong likelihood that the parameters will look significantly different at the end of the planning horizon than they did during design time. The robust supply chain design models in this dissertation address the need to incorporate uncertainty into the strategic planning process. In particular, we contribute to the literature on robust supply chain design as follows:

1. We develop a model to choose DC locations, assignments of retailers to DCs, and inventory levels at the DCs in order to minimize total expected systemwide cost. This model can also be used to model a multi-commodity supply chain design problem. We present a Lagrangian relaxation-based algorithm for this model.

2. We extend the first model by adding a constraint restricting the regret in any scenario to be no greater than a pre-specified robustness parameter, ensuring that solutions perform well both in the long run and in any individual scenario. We present another Lagrangian relaxation-based algorithm for this model. By systematically varying the robustness parameter, one can solve a minimax-regret version of the LMRP or, as a special case, of the uncapacitated fixed-charge location problem (UFLP). Since our algorithm may not perform well for certain values of the

robustness parameter, this method serves as a heuristic for the minimax-regret problem.

3. We present formulations of regret-constrained versions of the P-median problem (PMP) and the UFLP that are tighter than those from the relaxation described in number 2. These formulations are solved via variable-splitting.

No facility location models have been published that explicitly account for reliability issues—the ability of the system to perform well when not all facilities are operational. We make the following contributions:

4. We develop a model for choosing facility locations to minimize total fixed and transportation cost while restricting the maximum failure cost (the cost that results when a facility fails) to be no greater than a pre-specified value. We present several formulations and relaxations of this problem, as well as a tabu search heuristic for solving it. We show how to generate the tradeoff curve between operating cost and maximum failure cost using the constraint method of multi-objective optimization.

5. We develop a model for choosing facility locations to minimize a weighted sum of the operating cost (fixed plus transportation cost) and the expected failure cost. We solve this problem using Lagrangian relaxation and show how to generate the tradeoff curve between the two objectives using the weighting method of multi-objective optimization.

1.3 Outline

The remainder of this dissertation is organized as follows. In Chapter 2 we review the literature on robust and reliable supply chain design and facility location; we also discuss location–inventory models, focusing especially on the LMRP. In Chapter 3 we formulate and solve a stochastic version of the LMRP, present computational results, and discuss a multi-commodity interpretation of this model. In Chapter 4 we extend the stochastic problem by adding a constraint on the maximum regret, discuss the solution method, and resolve a number of computational difficulties arising from the additional constraint. We present tighter formulations of regret-constrained versions of the PMP and UFLP, and we present computational results. We discuss maximum failure cost reliability location problems in Chapter 5 and expected failure cost problems in Chapter 6. Both chapters include formulations, solution methods, and computational results. Finally, we summarize and discuss areas of future research in Chapter 7.

Chapter 2

Literature Review

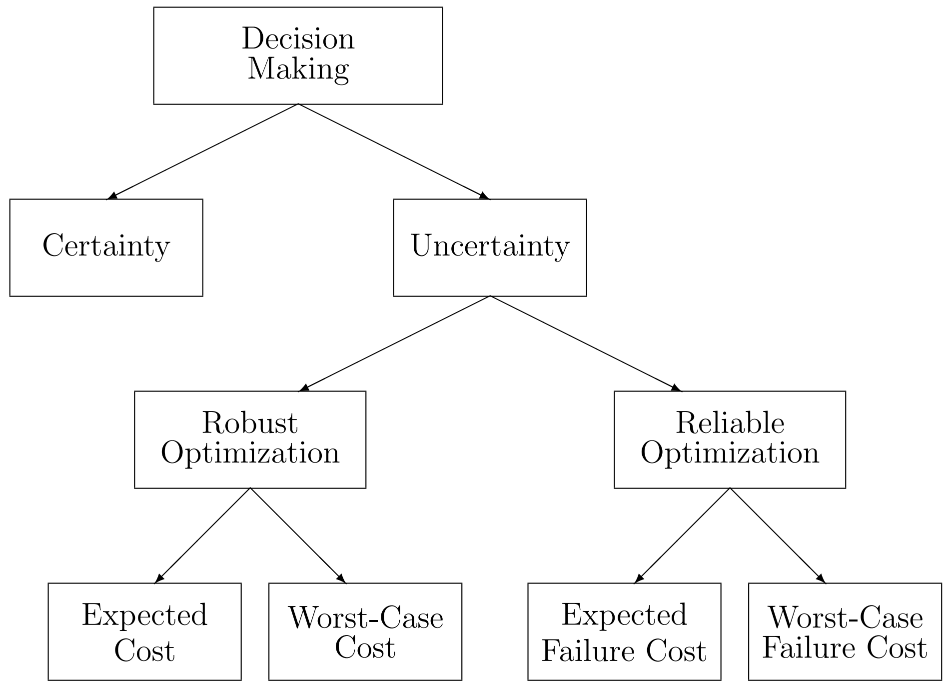
2.1 Introduction

Many authors divide decision-making environments into three categories: certainty, risk, and uncertainty (Rosenhead, Elton, and Gupta 1972). In *certainty* situations, all parameters are deterministic and known, whereas risk and uncertainty situations both involve randomness. In *risk* situations, there are uncertain parameters whose values are governed by probability distributions, which are known by the decision maker. In *uncertainty* situations, parameters are uncertain, and furthermore, no information about probabilities is known. In both situations, the goal is generally to find solutions that perform well under a variety of realizations of the uncertain parameters. Problems in risk situations are known as *stochastic optimization problems*, while problems under uncertainty are known as *robust optimization problems*. Stochastic optimization problems generally optimize the expected value of some objective function, while robust optimization problems involve guaranteeing that the solution chosen will perform well no matter what future comes to pass.

In this dissertation, the more relevant dichotomy is between *robust* and *reliable* optimization, which we defined in Chapter 1. Both types of optimization fall under the broad category of decision-making under uncertainty. (We do not reserve the word "uncertainty" to refer only to situations in which no probability information is known.) Thus, we suggest the following taxonomy of decision making (see Figure 2.1). At the highest level, decision-making problems take place in either certainty or uncertainty situations. This dissertation deals almost exclusively with the latter. Uncertainty situations can be classified as either robust or reliable optimization, with robust optimization dealing with uncertainty in data and reliable optimization dealing with uncertainty in solutions. Finally, robust optimization problems generally have an objective of minimizing either expected cost or worst-case cost (or regret), or a number of other measures that have been proposed in the literature and that are discussed below. Similarly, reliable optimization problems hedge against either expected cost of failure or worst-case cost due to failure.

This chapter has four main sections. In the first, we review several approaches to robust optimization, illustrating each with location modeling applications that have appeared in the literature. Although few models have been proposed that explicitly address reliable supply chain design, the concept bears some similarity to three previous streams of research, which we review briefly in Section 2.3. In Section 2.4, we review solution techniques that have been proposed for the capacitated facility location problem (CFLP), as some of the concepts used by previous authors are related to the algorithms developed

Figure 2.1: Decision-making taxonomy.



in this dissertation. Finally, the last section is devoted to location–inventory models, focusing especially on the Location Model with Risk-Pooling (LMRP).

Throughout this chapter we assume the reader has some familiarity with (deterministic) facility location theory. For an introduction into this topic, the reader is referred to the texts by Daskin (1995), Drezner (1995), or Hurter and Martinich (1989).

2.2 Robust Location Problems

The goal of robust optimization is to find a solution that will perform well under any possible realization of the random parameters. The definition of "performing well" varies from application to application and choosing an appropriate measure of robustness is part of the modeling process. Examples include minimizing expected and worst-case cost. The random parameters can be either continuous, in which case they are generally assumed to be statistically independent of one another, or described by discrete scenarios. If probability information is known, uncertainty is described using a (continuous or discrete) probability distribution on the parameters. If no probability information is known, parameters are generally described by intervals in the continuous case. The scenario approach has two main drawbacks. One is that identifying scenarios (let alone assigning probabilities to them) is a daunting and difficult task; indeed, it is the focus of an entire branch of stochastic programming theory. The second problem is that one generally wants to identify a relatively small number of scenarios for computational reasons, but this limits the range of options under which decisions are evaluated. But the scenario approach generally results in more tractable models, and furthermore, it has the advantage of allowing parameters to be statistically dependent, which is generally not practical in the continuous parameter approach. Dependence is often necessary to model reality, since, for example, demands are often correlated across time periods or geographical regions and costs are often correlated among suppliers.

In general, the robust location models discussed in this section are NP-hard. Min-expected-cost extensions of distance-based location models like the UFLP and P-median problem (PMP)—for example, those discussed in Section 2.2.1.2 below—are relatively easy to solve since they can often be treated as larger instances of deterministic problems. For example, a problem with 100 nodes and 10 scenarios can be solved in approximately the time required to solve a deterministic problem with 1000 nodes (several CPU minutes using today's state-of-the-art algorithms on a desktop computer). On the other hand, problems with a minimax structure like those discussed in Section 2.2.2 are more difficult to solve to optimality; today's best algorithms are able to solve problems perhaps an order of magnitude smaller than corresponding stochastic problems. This discrepancy parallels the difference in difficulty between distance-based and minimax deterministic problems. For example, relatively large instances of the UFLP and PMP may be solved quickly, but the P-center problem, which has a minimax structure, is generally solved by embedding a set-covering problem (which is itself NP-hard) inside a binary search routine.

2.2.1 Stochastic Location Problems

In this section, we discuss stochastic models for facility location. Most (though not all) of these models have as an objective to minimize the expected cost of the system. These models are solved using either special-purpose algorithms or more general stochastic programming techniques, which we review below. More comprehensive surveys on facility location under uncertainty are contained in Owen and Daskin (1998), ReVelle and Williams (2001), and Berman and Krass (2001). Daskin and Owen (1999) and Current, Daskin, and Schilling (2001) overview both deterministic and stochastic facility location. See the textbook by Birge and Louveaux (1997) for an introduction to stochastic programming techniques.

Probably the first attempt to solve location problems under uncertainty is that of Frank (1966), who extends the work of Hakimi (1964, 1965) to consider probabilistic centers and medians on a graph with independent random demands. Frank presents methods for finding "max probability" centers (points that maximize the probability that the maximum weighted distance from the point is within a given limit) and medians (points that maximize the probability that the total demand-weighted distance from the point is within a given limit). Frank (1967) extends this analysis to jointly distributed normal demands.

Sheppard (1974) was one of the first authors to propose a scenario approach to facility location. He suggests selecting facility locations to minimize the expected cost, though he does not discuss the issue at length. In any stochastic programming problem, one must determine which decision variables are first-stage and which are second-stage; that is, which variables must be set now and which may be set after the uncertainty has been resolved. In stochastic location modeling, locations are generally first-stage decisions while assignments of customers to facilities are second-stage decisions. (If both decisions occur in the first stage, most problems can be reduced easily to deterministic problems in which uncertain parameters are replaced by their means.)

2.2.1.1 Analytical Properties

Several of the early stochastic location papers were devoted to establishing whether the Hakimi property holds. The Hakimi property (Hakimi 1964, 1965) states that there exists an optimal solution to a network location problem in which the facilities are located on the nodes of the network, not along the edges. Mirchandani and Odoni (1979) prove the Hakimi property for a PMP on a network with shortest-path travel costs in which the cost of a path may be any concave, non-decreasing function of its length, under a mild homogeneity assumption. In their problem, both demands and transportation costs may be uncertain. Mirchandani (1980) uses similar analysis to determine whether the Hakimi property applies to stochastic versions of the PMP and uncapacitated fixed-charge location problems (UFLP; Balinski 1965) under a variety of assumptions. Louveaux and Thisse (1985) maximize expected utility of profit in a production–distribution system in which they locate a single facility and set production levels in the first stage and make distribution decisions in the second; they show that the Hakimi property applies when the firm is risk-neutral (i.e., the utility function is linear) but not when it is risk-averse.

2.2.1.2 Special-Purpose Algorithms

Weaver and Church (1983) present a Lagrangian relaxation algorithm for the stochastic PMP discussed by Mirchandani and Odoni (1979). Their formulation and algorithm are similar to ours for the model presented in Chapter 3, so we will discuss it in a bit of detail. Imagine a P-median problem with customer set I and facility set J. Demands and distances are random, determined by scenarios. Let S be the set of scenarios, q_s the probability that scenario s occurs, h_{is} the demand of customer i in scenario s, and d_{ijs} the distance from customer i to facility location j in scenario s. Define decision variables

$$X_j = \begin{cases} 1, & \text{if facility location } j \text{ is chosen to be in the solution,} \\ 0, & \text{otherwise} \end{cases}$$

for $j \in J$, and

$$Y_{ijs} = \begin{cases} 1, & \text{if a facility at location } j \text{ serves customer } i \text{ in scenario } s, \\ 0, & \text{otherwise} \end{cases}$$

for $i \in I,\ j \in J, s \in S$. The stochastic P-median problem can be formulated as follows:

$$\text{(SPMP)} \qquad \text{minimize} \qquad \sum_{s\in S}\sum_{i\in I}\sum_{j\in J} q_s h_{is} d_{ijs} Y_{ijs} \tag{2.1}$$

$$\text{subject to} \qquad \sum_{j\in J} Y_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{2.2}$$

$$Y_{ijs} \leq X_j \qquad \forall i \in I, \forall j \in J, \forall s \in S \tag{2.3}$$

$$\sum_{j\in J} X_j = P \tag{2.4}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{2.5}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall s \in S \tag{2.6}$$

The objective function (2.1) minimizes the expected demand-weighted distance; constraints (2.2) require each customer to be assigned to exactly one facility in each scenario; constraints (2.3) prohibit a customer from being assigned to a facility that has not been opened; constraint (2.4) requires exactly P facilities to be opened; and constraints (2.5) and (2.6) are standard integrality constraints.

Weaver and Church's solution method for (SPMP) is to relax the assignment constraints (2.2), resulting in the following Lagrangian subproblem:

$$\text{(LR-WC)} \quad \text{maximize}_\lambda \quad \text{minimize}_{X,Y} \quad \sum_{s\in S}\sum_{i\in I}\sum_{j\in J}(q_s h_{is} d_{ijs} - \lambda_{is})Y_{ijs} + \sum_{s\in S}\sum_{i\in I}\lambda_{is} \tag{2.7}$$

subject to constraints (2.3)–(2.6), where λ_{is} is the Lagrange multiplier corresponding to constraint (2.2) for customer i and scenario s. This problem can be solved easily. Suppose X_j is set to 1. For a given $i \in I$ and $s \in S$, Y_{ijs} will be set to 1 if and only if $q_s h_{is} d_{ijs} - \lambda_{is} < 0$. We can compute the *benefit* of opening facility j (i.e., the contribution to the objective function if j is opened) as

$$\gamma_j = \sum_{s \in S} \sum_{i \in I} \min\{0, q_s h_{is} d_{ijs} - \lambda_{is}\}.$$

To solve (LR-WC), we set $X_j = 1$ for the P facilities with smallest γ_j, and set $Y_{ijs} = 1$ if $X_j = 1$ and $q_s h_{is} d_{ijs} - \lambda_{is} < 0$. The optimal objective value for (LR-WC) provides a lower bound on the optimal objective value for (SPMP). Upper bounds are found using a heuristic at each iteration, and the Lagrange multipliers are updated using subgradient optimization. If the Lagrangian procedure terminates with the lower bound not equal to the upper bound, branch-and-bound can be used to close the gap.

Essentially, Weaver and Church's method treats the problem as a deterministic PMP with $|I||S|$ customers instead of $|I|$ and then applies the standard Lagrangian relaxation method for it (Cornuejols, Fisher, and Nemhauser 1977). We use a similar idea in Chapter 3, but our more complicated problem prohibits all of the customers from being lumped together when computing the benefits; instead, they must be grouped by scenario.

Mirchandani, Oudjit, and Wong (1985) begin with the same formulation as Weaver and Church, explicitly reformulating it as a deterministic PMP with $|I||S|$ customers contained in a new customer set I'. Each customer in I' corresponds to a customer–scenario pair in the original problem. If customer $i' \in I'$ corresponds to (i, s) in the (SPMP), then $h_{i'} = q_s h_{is}$ and $d_{i'j} = d_{ijs}$ for $j \in J$. Like Weaver and Church, Mirchandani, et al. also suggest a Lagrangian relaxation method, but instead of relaxing the assignment constraints, they relax the single P-median constraint (2.4). The resulting subproblem is:

$$\text{(LR-MOW)} \quad \text{maximize}_f \quad \text{minimize}_{X,Y} \quad \sum_{i' \in I'} \sum_{j \in J} h_{i'} d_{i'j} Y_{i'j} + f\left(\sum_{j \in J} X_j - P\right) \tag{2.8}$$

subject to constraints (2.2), (2.3), (2.5), and (2.6). Ignoring the constant $-fP$ in the objective function, this problem is a UFLP in which all facilities have the same fixed cost f. The authors solve this subproblem using Erlenkotter's (1978) DUALOC algorithm and update the multiplier f using a subgradient-type method. They prove that the procedure is guaranteed to converge to the optimal f in no more than $|I|-1$ iterations; moreover, since the UFLP does not have the integrality property, the optimal objective of (LR-MOW) is at least as large as the optimal objective of (LR-WC), meaning this method provides a tighter lower bound (see Section 2.4). Computational results for this method are very promising. It cannot be applied to our problems, however, since they are based on the UFLP, rather than on the PMP; there is no P-median constraint to relax in our models. The authors refer to their algorithm as a "nested dual approach" since it involves first forming the Lagrangian dual and then solving it using Erlenkotter's dual-based method.

Berman and LeBlanc (1984) study a problem in which travel times are stochastic and facilities may be relocated (at a cost) as conditions change. The objective is to minimize expected transportation and relocation costs. The authors present a polynomial-time heuristic that loops through the states, performs local exchanges within each state, and then performs exchanges to tie the states to each other better in order to reduce relocation costs. Carson and Batta (1990) present a case study of a similar problem in which they relocate a single ambulance on SUNY Buffalo's Amherst campus as the population moves about the campus throughout the day (from classroom buildings to dining halls to dorms, etc.). The authors point out that papers like Berman and LeBlanc's are difficult to apply in practical settings because they require knowledge of utility functions, probability densities, and so on. Instead, Carson and Batta simply divide the day into four unequal time periods and solve a 1-median problem in each. Relocation costs are not explicitly considered, but the decision to use four time periods was arrived at as a tradeoff between frequent relocation and increased response times. We wish to point out that the LMRP and its stochastic extensions presented in this dissertation contain a large number of parameters, but with few exceptions, these can be ignored (set to 0 or made equal for each retailer and/or scenario), if desired. We attempt, whenever possible, to avoid the data-requirement trap described by Carson and Batta.

2.2.1.3 Stochastic Programming Methods

Over the past few decades, the field of stochastic programming has become increasingly well developed. The two-stage nature of many facility location problems (locate in the first stage while parameters are still random, and make assignment or transportation decisions in the second stage when uncertainties have been resolved) has made location a popular application of general stochastic programming methods (as opposed to the special-purpose models and algorithms described thus far).

Carbone (1974) uses chance-constrained programming to make facility location decisions when demands are normally distributed, possibly correlated. The problem he considers is to locate P facilities to minimize an upper bound on the total demand-weighted distance that can be achieved with probability α; that is, to minimize K such that

$$P\left(\sum_i \sum_j h_i d_{ij} Y_{ij} \leq K\right) \leq \alpha,$$

where h_i is the (random) demand of customer i, d_{ij} is the distance from customer i to facility j, Y_{ij} is 1 if we assign customer i to facility j, and $0 \leq \alpha \leq 1$ is a constant.

Louveaux (1986) presents stochastic versions of the capacitated P-median problem and capacitated fixed-charge location problem (CFLP) in which demand, production costs, and selling prices are random. The goal is to choose facility locations, determine their capacities, and decide which customers to serve and from which facilities to maximize the expected utility of profit. Since demands are random and facilities are capacitated, the facilities chosen in the first stage may be insufficient to serve all of the demands in the second stage; hence a penalty for unmet demand is included in the models. To formulate the stochastic capacitated PMP, the constraint requiring P facilities to be opened is replaced by a budget constraint on the total cost (fixed cost, capacity cost, and transportation cost) that must be satisfied under any realization of the demand; the budget can be used to determine P. The author shows that under a particular type of budget constraint, the two stochastic models (CFLP and capacitated PMP) are equivalent. Louveaux and Peeters (1992) present a dual-based heuristic for the CFLP model presented in Louveaux (1986), and Laporte, Louveaux, and van Hamme (1994) present an optimal algorithm.

França and Luna (1982) use Benders decomposition to solve a problem that is a combination of the CFLP and the stochastic transportation problem with random demands. In the first stage, the firm chooses facility locations and decides how much to ship to each demand point, and in the second stage it meets the demand, possibly incurring shortage or holding costs. Jornsten and Bjorndal (1994) choose where and when to locate facilities over time in order to minimize the expected time-discounted cost; production and distribution costs are random. Their algorithm uses scenario aggregation and an augmented Lagrangian approach and is practical for small problems. Eppen, Martin, and Schrage (1989) solve a multi-period capacity planning problem with random demand and selling prices; capacity levels are set in the first stage and production levels in the second. The objective is to choose capacity configurations at each plant in each time period, subject to a re-tooling cost for changing capacity, to maximize the expected time-discounted profit subject to a limit on expected downside risk (EDR). Their algorithm involves successively tightening the EDR constraint and re-solving, resulting in multiple solutions; a decision maker can choose among these solutions based on the tradeoff between expected profit and EDR. The formulation presented in the paper is solved by a general-purpose MIP solver but is very large—some parameters and variables have as many as five subscripts—making it practical only for small problems.

2.2.1.4 Economic Models

Some of the earlier works on facility location under uncertainty used economic theory to determine optimal facility locations or sizes. Recognizing that different firms may wish to treat uncertainty differently, many of these models concentrate on incorporating a firm's level of risk aversion into the decision-making process. Jucker and Carlson (1976) use a mean–variance objective function in a stochastic formulation of the uncapacitated fixed-charge location problem (UFLP) in which selling price (and hence demand) may be random. They present solution methods for four types of firms, all risk-averse, characterized by which variables (e.g., price) they set and which others (e.g., demand) they accept as a result. Hodder and Jucker (1985) extend Jucker and Carlson's model to allow for correlation among the random prices. Their model is a quadratic programming problem but can be solved easily. Hodder (1984) incorporates the capital asset pricing model (CAPM) objective into the UFLP and compares it to the mean–variance objective. Manne (1961) uses economic arguments to determine plans for capacity expansion under uncertainty when future demands follow an upward trend with a random walk. His work shows that it is sufficient to replace the randomness with its deterministic trend. More recently, his work was extended by Bean, Higle, and Smith (1992), who use stochastic programming theory to show similar results. Verter and Dincer (1992) review the literature on stochastic facility location and capacity expansion problems, focusing particularly on economics-based papers such as these. Hanink (1984) uses portfolio theory to solve location problems, while Blanchini, Rinaldi, and Ukovich (1997) use game theory. Cheung and Powell (1996) evaluate the attractiveness of a set of facility locations by determining the competitive equilibrium that would result and the firm's consequent market share.

Cheung and Powell's paper is related to another group of papers that seeks to evaluate the "option value" of a given set of worldwide facility locations. Constructing facilities in several countries gives a firm extra flexibility since it can shift production to countries with favorable exchange rates, local costs, labor availability, etc. Huchzermeier and Cohen (1996) evaluate "operational options" (choices for a logistics network) over multiple time periods under uncertainty in exchange rates. They build a Markov model of exchange rates and solve a supply chain design problem for each scenario, then use stochastic dynamic programming to determine the value of each option. Kogut and Kulatilaka (1994) similarly use dynamic programming to evaluate options when there is a cost for switching production from one site to another. They discuss the threshold at which switching becomes advantageous and make the observation that the model favors countries with volatile exchange rates since they provide greater opportunity to take advantage of fluctuations. This counter-intuitive result illustrates the difference between financial and operational hedging: while financial hedging seeks to eliminate volatility in cash flows, operational hedging seeks to exploit it. Lowe, Wendell, and Hu (1999) provide a decision-analysis approach for the operational hedging concept, illustrating its use with a popular Harvard Business Review case. Gutiérrez and Kouvelis (1995) present a model to choose suppliers internationally to hedge against changes in exchange rates and local costs; unlike the first three models described in this paragraph, which are descriptive, Gutiérrez and Kouvelis's model is normative, though necessarily less rich than the descriptive models. We discuss their model further in Section 2.2.4.

2.2.2 Minimax Location Models

When no probability information is known, the expected cost measure is irrelevant. Many measures of robustness have been proposed in the literature for this situation. The two most common are minimax cost and minimax regret, which are closely related to one another and are discussed in this section. Other less common measures are discussed in Section 2.2.3. The primary robustness measures are discussed in the recent text on robust optimization by Kouvelis and Yu (1996), though they use somewhat different terminology than we do. The *minimax cost solution* is the one that minimizes the maximum cost across all scenarios. This measure is, on the one hand, overly conservative, emphasizing the worst possible scenario, and on the other hand, somewhat reckless, since it may produce quite poor solutions for scenarios other than the one with maximum cost, especially if the scenarios have a form like "small demand / moderate demand / large demand." It may be appropriate for situations in which, for example, a firm's competitors will make decisions that make the worst scenario come to pass for the firm.

The other two most common robustness measures consider the *regret* of a solution, which is the difference (absolute or percentage) between the cost of a solution in a given scenario and the cost of the optimal solution for that scenario. Models that seek to minimize the maximum (absolute or relative) regret across all scenarios are called *minimax (absolute or relative) regret* models. Minimax regret models are commonly employed in the literature. Generally such problems are solved using problem-specific algorithms, though a general-purpose heuristic for minimax regret linear programs with interval data was proposed by Mausser and Laguna (1999a) and an exact algorithm for the same problem was proposed by Mausser and Laguna (1999b). The optimal algorithm (1999b) relies on the fact that for a given solution, each parameter is set to either its lower or upper endpoint in the regret-maximizing scenario. To identify this scenario, the authors solve a MIP that adds five constraints and one binary variable to the original problem for each uncertain parameter; this approach is practical for small- to moderate-sized LPs. The heuristic (1999a) is a greedy heuristic that contains some methods for diversification to avoid local optima. It can be used on its own or in place of the exact solution to the MIP described in (1999b). Minimax cost problems can often be transformed into minimax regret problems, and vice-versa, since the cost and regret of a given scenario differ by a constant. Solution approaches for one criterion are often applicable to the other, as well.

The regret criterion is usually applied in uncertainty situations. It has been discussed in the context of risk situations as well, but minimizing expected regret is equivalent to minimizing expected cost. To see this, consider a general min-expected-regret problem with variables $x_1, \ldots, x_n$, feasible set X, scenarios $s \in S$, objective function coefficients c_{is}, scenario probabilities q_s, and optimal scenario objective values z_s^*.

$$\text{minimize} \quad \sum_{s \in S} q_s R_s \tag{2.9}$$

$$\text{subject to} \quad R_s = \sum_{i=1}^{n} c_{is} x_i - z_s^* \qquad \forall s \in S \tag{2.10}$$

$$x \in X \tag{2.11}$$

Substituting the regret variables R_s into the objective function, we get

$$\text{minimize} \quad \sum_{s \in S} q_s \left(\sum_{i=1}^{n} c_{is} x_i - z_s^* \right) \tag{2.12}$$

$$\text{subject to} \quad x \in X \tag{2.13}$$

The objective function of this revised problem is the min-expected-cost objective function minus a constant. (This equivalence is sometimes overlooked in the literature.)

Regret-based problems tend to be more difficult than stochastic problems because of their minimax structure. On the other hand, they lend themselves more easily to analytical results, frequently in limited contexts such as 1-median problems or P-medians on tree networks. For example, Chen and Lin (1998) present a polynomial-time algorithm for the 1-median problem on a tree with random, interval-based edge lengths and node weights. As in many minimax problems, the Hakimi property does not apply to this problem. In Chen and Lin's problem, node weights must be non-negative; Burkhard and Dollani (2001) present a polynomial algorithm for the case in which node weights can be positive or negative. Vairaktarakis and Kouvelis (1999) similarly consider 1-medians on a tree, but in their problem, edge lengths and node weights may be linear over time (i.e., not stochastic but deterministic and dynamic) or random and scenario-based. They trace the path of the solution over time (in the dynamic case) and present low-order polynomial algorithms for both cases. Averbakh and Berman (2000) consider the minimax regret 1-median on a general network with random, interval-based demands. They present the first polynomial-time algorithms for the problem on a general network and present algorithms for tree networks that have lower complexity than those previously published.

Averbakh and Berman (1997) consider the minimax regret weighted P-center problem on a general network with uncertain, interval-based demands. (The deterministic weighted P-center problem is to locate P facilities to minimize the maximum weighted distance traveled by any customer to its nearest facility.) They show that the minimax regret problem can be solved by solving $n+1$ deterministic weighted P-center problems: n of them on the original network and 1 on an augmented network, where n is the number of nodes in the problem. Since the weighted P-center problem can be solved in polynomial time for the special cases in which $P = 1$ or the network is a tree, this leads to a polynomial-time algorithm for the minimax problem in these cases.

In many minimax regret papers, the general strategy of the algorithm is as follows:

1. Choose a candidate solution x.

2. Determine the maximum regret across all scenarios if solution x is chosen. For scenario-based uncertainty, this is easy: just compute the cost of the solution under each scenario and compare it to the optimal cost for the scenario, then choose the scenario with the greatest regret. For interval-based uncertainty, techniques for finding the regret-maximizing scenario rely on the fact that this scenario typically has all parameters set to an endpoint of their intervals. Still, this problem can be quite difficult. Solving this problem is the crux of the algorithms by Mausser and Laguna (1999a, 1999b), discussed above. On the other hand, Averbakh and Berman (2000) develop an $O(n^2)$ algorithm to determine the regret-maximizing scenario for their problem.

3. Either repeat steps 1 and 2 for all possible solutions (as in Averbakh and Berman 2000), or somehow find a new candidate solution whose regret is smaller than the regret determined in step 2 (as in Mausser and Laguna 1999b).

We now turn our attention to problems with scenario-based uncertainty on general networks. Serra, Ratick, and ReVelle (1996) solve the maximum capture problem (to locate P facilities in order to capture the maximum market share, given that the firm's competitors have already located their facilities) under scenario-based demand uncertainty. They consider both maximizing the minimum market share captured (the maximization analog of the "minimax cost" criterion described above) and minimizing maximum regret. They present a heuristic that involves solving the deterministic problem for each scenario, choosing an initial solution based on those results, and then using an exchange heuristic to improve the solution. A similar approach is used by Serra and Marianov (1998), who solve the minimax cost and minimax regret problems for a P-median problem, also under scenario-based demand uncertainty. They present a case study involving locating fire stations in Barcelona. In the model presented by Current, Ratick, and ReVelle (1997), facilities are located over time, but the number of facilities that will ultimately be located is uncertain. The model is called NOFUN (number of facilities uncertain). The approach is scenario-based (scenarios dictate the number of facilities to open), and the authors discuss the objectives of both minimizing expected regret and minimizing maximum regret. The authors' proposed formulation is based on the PMP and is solved using a general-purpose MIP solver.

Not all deterministic problems that are polynomially solvable have robust versions that are polynomially solvable. For example, the economic order quantity (EOQ) model is still easy in its minimax regret form (Yu 1997), but the minimax regret shortest path problem is NP-hard (Yu and Yang 1998). Daniels and Kouvelis (1995) solve a minimax regret version of a machine scheduling problem whose deterministic form is easy. Their algorithm follows the general form given above. For a given solution x, finding the regret-maximizing scenario in step 2 turns out to be an assignment problem. Given some bounds on the regret, finding a candidate solution in step 1 is done using *surrogate relaxation* (Glover 1975). The basic idea is that by replacing the regret constraints with their weighted sum, one obtains a deterministic scheduling problem whose solution can be found quickly using the shortest-processing-time-first (SPT) rule. By changing the weights systematically, we tighten the bounds that this problem provides.

2.2.3 Other Robustness Measures

2.2.3.1 Robustness and Stability

Several other robustness measures have been proposed. One of the earliest was proposed by Gupta and Rosenhead (1968) and Rosenhead, Elton, and Gupta (1972). In these papers, decisions are made over time, and a solution is considered more robust if it precludes fewer good outcomes for the future. An example in the latter paper concerns a facility location problem in which a firm wants to locate five facilities over time. Suppose all possible five-facility solutions have been enumerated, and N of them have cost less than or equal to some pre-specified value. If facility j is included in p of the N solutions, then its *robustness* is p/N. One should construct the more robust facilities first, then make decisions about future facilities as time elapses and information about uncertain parameters becomes known. Now suppose that the first facility has been constructed and the firm decides (because of budget, politics, shrinking demand, etc.) not to build any of the other facilities. The *stability* of a facility is concerned with how well the facility performs if it is the only one operating. Stability should be used to distinguish among facilities that are nearly equally robust. Note that these definitions of robustness and stability refer to individual facilities, not to solutions as a whole.

This robustness criterion is dissatisfying because it considers only decisions that evolve over time and says little about decisions that must be made now but perform well in the future. In addition, computing the measure requires enumerating all possible solutions, which is generally not practical. Therefore, this measure has not been used much. Schilling (1982) presents two location models that use this robustness measure, both using stochastic, scenario-based demands. The first model is a set-covering-type model that maximizes the number of facilities in common across scenarios subject to all demands being covered in all scenarios and a fixed number of facilities being located in each scenario. By varying this last parameter, one can obtain a tradeoff curve between the total number of facilities constructed and the number of facilities that are common across scenarios. If the firm is willing to build a few extra facilities, it may be able to substantially delay the time until a single solution must be chosen, since the common facilities can be built first. The second model is a max-covering-type model that maximizes the coverage in each scenario subject to the number of common facilities exceeding some threshold. In this case the tradeoff curve represents the balance between demand coverage and common facilities. Unfortunately, Schilling's models were shown by Daskin, Hopp, and Medina (1992) to produce the worst possible results in some cases. To see why, imagine a firm that wants to locate two distribution centers (DCs) to serve its three customers, in New York, Boston, and Spokane. New York has either 45% or 35% of the demand and Boston has 35% or 45% of the demand, depending on the scenario. The remaining 20% of the demand is in Spokane, in either scenario. If the transportation costs are sufficiently large, the optimal solution in scenario 1 is to locate in New York and Spokane, while the optimal solution in scenario 2 is to locate in Boston and Spokane. Current's method would instruct the firm to build a DC in Spokane first, since that location is common to both solutions, then wait until some of the uncertainty is resolved before choosing the second site. But then all of the east-coast demand is served from Spokane for a time—clearly a suboptimal result.

Rosenblatt and Lee (1987) use a similar robustness measure to solve a facility layout problem. Unlike Rosenhead et al.'s measure, which considers the percentage of good solutions that contain a given *element* (e.g., facility), Rosenblatt and Lee consider the percentage of scenarios for which a given *solution* is "good," i.e., has regret bounded by some pre-specified limit. Like the previous measure, Rosenblatt and Lee's measure requires enumerating all solutions and evaluating each solution under every scenario, making this measure practical only for very small problems.

2.2.3.2 Model and Solution Robustness

Mulvey, Vanderbei, and Zenios (1995) introduce a new framework for robust optimization (RO). Their framework involves two types of robustness: *solution robustness* (the solution is "nearly" optimal in all scenarios) and *model robustness* (the solution is "nearly" feasible in all scenarios). The definition of "nearly" is left up to the modeler; their objective function has very general penalty functions for both model and solution robustness, weighted by a parameter intended to capture the modeler's preference between the two. The solution robustness penalty might be the expected cost, maximum regret, or von Neumann–Morgenstern utility function. The model robustness penalty might be the sum of the squared violations of the constraints. Uncertainty may be represented by scenarios or intervals, with or without probability distributions. The authors discuss a number of applications in which the RO framework has been applied. In one example, a power company wants to choose the capacities of its plants to minimize cost while meeting customer demand and satisfying certain physical constraints. In the RO model for this problem, the objective function has the form

$$\text{minimize } E[\text{cost}] + \lambda \text{Var}[\text{cost}] + \omega[\text{sum of squares of infeasibilities}].$$

The first two terms represent solution robustness, capturing the firm's desire for low costs and its degree of risk-aversion, while the third term represents model robustness, penalizing solutions that fail to meet demand in a scenario or violate other physical constraints like capacity.

Because of the flexibility of the general RO model, we cannot expect to develop algorithms that will solve every RO problem; algorithms will have to be somewhat problem-specific. This makes the RO approach somewhat limited. Nevertheless, in the eight years since Mulvey et al.'s paper was published, it has received a great deal of attention in the literature. A recent citation search revealed over 50 articles citing their work. In part, this is due to the generality of their model—nearly any stochastic or robust optimization model can fit the RO framework. But the attention is also due to the fact that researchers have increasingly begun to recognize the importance of robustness in a wide variety of applications. The RO framework is explicitly used in applications as varied as parallel machine scheduling with stochastic interruptions (Laguna et al. 2000), relocation of animal species under uncertainty in population growth and future funding (Haight, Ralls, and Starfield 2000), production planning (Trafalis, Mishina and Foote 1999), large-scale logistics systems (Yu and Li 2000), and chemical engineering (Darlington et al. 2000).

Killmer, Anandalingam, and Malcolm (2001) use the RO framework to find solution- and model-robust solutions to a stochastic noxious facility location problem.[1] The RO model for this problem minimizes the expected cost plus penalties for regret, unmet demand, and unused capacity. The expected cost and regret penalty are the solution robustness terms (encouraging solutions to be close to optimal), while the demand and capacity violation penalties are model robustness terms (encouraging solutions to be close to feasible). The non-linear programming model is applied to a small case study in Albany, NY and is solved using MINOS.

2.2.3.3 Restricting Outcomes

One use of the model robustness term in the RO model is to penalize solutions for being too different across scenarios (in terms of variables, not costs), thus encouraging the resulting solution to be insensitive to uncertainties in the data. Vladimirou and Zenios (1997) formulate several models for solving this particular realization of the RO framework, which they call *restricted recourse*. Restricted recourse in itself is a valid and interesting robustness measure. It might be appropriate, for example, in a production planning context in which re-tooling in each period is costly. However, there may be a substantial tradeoff between robustness (in this sense) and cost. The authors present three procedures for solving such problems, each of which begins by forcing all second-

[1] Though the authors discuss their model solely in the context of noxious facility location, it is similar to the UFLP and could be applied to much broader problems than noxious facility location.

stage solutions to be equal, and then gradually loosens that requirement until a feasible solution is found. The stochastic programming problems are solved using standard integer SP algorithms. The authors analyze the trade-off between robustness and cost, and often find large increases in cost as the restricted recourse constraint is made more stringent.

In contrast, Paraskevopoulos, Karakitsos, and Rustem (1991) present a model for robust capacity planning in which they restrict the sensitivity of the *objective function* (rather than the variables) to changes in the data. Instead of minimizing expected cost, Paraskevopoulos et al. minimize expected cost plus a penalty on the objective's sensitivity to changes in demand. The penalty is weighted based on the decision-maker's level of risk aversion. The advantage of this robustness measure is that the resulting problem looks like the deterministic problem with the uncertain parameters replaced by their means and with an extra penalty term added to the objective. Scenarios and probability distributions do not enter the mix. The down-side is that computing the penalty requires differentiating the cost with respect to the error in the data. For realistic capacity-planning problems, even computing the expected cost (let alone its derivative) is difficult and in some cases must be done via Monte Carlo simulation. For linear models, including most location models, computing the expected cost may be easy, but the penalty becomes a constant and the problem reduces to the deterministic problem in which uncertain parameters are replaced by their means; this generally gives poor results. Understandably, Paraskevopoulos et al.'s robustness measure has not been applied to location problems.

The requirement that solutions be similar across scenarios in terms of cost (rather than in terms of variables) has some similarity to the notion of p-robustness, defined below in Section 2.2.4.

2.2.3.4 α-Reliability

Another extension to the concepts described above was developed by Daskin, Hesse, and ReVelle (1997) and Owen (1999), who present the notion of *α-reliability*. The idea behind α-reliability is that the minimax cost and minimax regret criteria tend to focus on a few scenarios which may be catastrophic but are unlikely to occur. In the α-reliable framework, the robustness criterion of choice (say minimax regret) is applied only to a subset of scenarios, called the *reliability set*, whose total probability is at least α. Therefore, the probability that a scenario that was not included in the objective function comes to pass is bounded by $1-\alpha$. The parameter α is specified by the modeler but the reliability set is chosen endogenously. The example given in the paper applies the α-reliability concept to the minimax-regret P-median problem. The problems are solved using standard LP/branch-and-bound techniques, though Owen (1999) develops a genetic algorithm to solve the problem. α-reliability can be thought of as a hybrid measure since it combines aspects of risk (scenario probabilities) and uncertainty (regret criteria). The robustness measure we present in Chapter 4 is also a hybrid measure, combining an expected cost objective with a constraint on regret.

2.2.4 p-Robustness

Kouvelis, Kurawarwala, and Gutiérrez (1992) present a new measure of robustness which involves a constraint dictating that the relative regret in any scenario must be no greater than p, where $p \geq 0$ is an external parameter. In other words, the cost under each scenario must be within $100(1+p)\%$ of the optimal cost for that scenario. We will refer to this measure as *p-robustness* throughout this dissertation, though Kouvelis et al. refer to it simply as "robustness." Note that for small p, there may be no p-robust solutions for a given problem. Thus p-robustness adds a feasibility issue not present in most other robustness measures.

The problem considered in Kouvelis et al. (1992) is a facility layout problem in which the goal is to construct a list of p-robust solutions, if they exist. The facility layout problem is modeled as a quadratic assignment problem (QAP), and the proposed algorithm is a modification of a standard branch-and-bound algorithm for the QAP. The problem is solved separately for each scenario, and each time a feasible solution is found in any of the branch-and-bound trees, its regret is computed for each scenario; if its maximum regret is less than or equal to p, the solution is added to a list of p-robust solutions. Nodes are fathomed from the branch-and-bound trees if $(1-p)\text{LB} > \text{UB}$, not if LB > UB as in the usual branch-and-bound method. This algorithm is dissatisfying for a number of reasons. First, there is no focused effort to find p-robust solutions—they are simply a by-product of the searches for individual-scenario solutions. Second, there is no guarantee that the resulting list of p-robust solutions is exhaustive, or even that a p-robust solution will be found if one exists. The method suffers from the paradoxical problem that it accomplishes its goal (finding p-robust solutions) best when the algorithm performs poorly, since more iterations mean more candidate solutions considered and more possible p-robust solutions. Third, there is no overall objective that helps a decision-maker distinguish among the p-robust solutions returned. The computational results indicate that as many as 400 p-robust solutions may be found for reasonable values of p. One solution to this problem is to reduce p; another is to rank the list in order of expected cost (if probabilities are available) or maximum regret. We will avoid all of these problems in our algorithm for the p-robust LMRP presented in Chapter 4.

The p-robust criterion is also used in two other papers: Gutiérrez and Kouvelis's (1995) paper on robust solutions for an international sourcing problem and Gutiérrez, Kouvelis, and Kurawarwala's (1996) paper on robust network design. All three papers are also discussed in Kouvelis and Yu's (1997) book. The international sourcing paper (Gutiérrez and Kouvelis 1995) presents an algorithm that, for a given p and N, returns either all p-robust solutions (if there are fewer than N of them) or the N solutions with smallest maximum regret. The sourcing problem considered involves choosing suppliers worldwide so as to hedge against changes in exchange rates and local prices. The problem reduces to the uncapacitated fixed-charge location problem, so the authors are essentially solving a p-robust version of the UFLP. Their algorithm maintains separate branch-and-bound trees for each scenario, and all trees are explored and fathomed simultaneously. Unfortunately, their algorithm contains an error. The authors implicitly make the faulty assumption that the child of a node in the branch-and-bound tree cannot have an optimal solution with smaller maximum regret than the solution at the node itself. This causes the tree to be fathomed inappropriately, resulting in sub-optimal solutions being returned. We discuss this problem in more detail in the Appendix.

The network design paper (Gutiérrez et al. 1996) uses Benders decomposition to search for a p-robust solution to an uncapacitated network design problem. Like the layout problem in Kouvelis et al. (1992), the problem in this paper is a feasibility problem only—no objective function is used to differentiate among p-robust solutions. The algorithm in this paper solves separate network design problems for each scenario, though they are linked by feasibility cuts that are added simultaneously to all problems; it suffers from the same problems as Kouvelis et al.'s (1992) algorithm.

2.3 Reliable Supply Chain Design

Though no models have been published to date that explicitly consider reliable supply chain design, there are three main bodies of literature that are similar in spirit, if not in modeling approach. The first is the literature on network reliability, most often applied to telecommunications or power transmission networks. In a typical network reliability problem, the edges (or, less frequently, the nodes) of a network are subject to failure with a given probability, and the goal is to maximize (or simply estimate) the probability that the network remains connected. The second body of literature concerns expected or backup covering models, which are frequently used in locating emergency services vehicles or facilities. Finally, our models can be seen as an outgrowth of a small body of literature that discusses approaches for handling disruptions to supply chains but presents few, if any, quantitative models. We discuss each of these three topics next.

2.3.1 Network Reliability

Network reliability theory is concerned with computing, estimating, or maximizing the probability that a network (typically a telecommunications or power network, represented by a graph network) remains connected in the face of random failures. (See the textbooks by Colbourn 1987, Shier 1991, or Shooman 2002.) Failures may be due to disruptions, congestion, or blockages. In almost all cases, failures occur only on the edges, but occasional papers consider node failures as well (e.g., Eiselt, Gendreau, and Laporte 1996). Various measures of post-failure connectivity have been considered; for example, two-terminal reliability (the probability that two given nodes s and t can communicate), all-terminal reliability (the probability that all nodes can communicate), and node pair resilience (the expected number of node pairs that can communicate).

The network reliability literature tends to focus either on computing reliability or on optimizing it, i.e., designing reliable systems. Computing the reliability of a given network is a non-trivial problem (see, e.g., Ball 1979), and various techniques have been proposed for computing or estimating the desired probabilities. These include cut-set and tie-set analysis (enumerating the cut-sets or tie-sets connecting the nodes of interest and computing the probability that the sets required for connectivity remain in place) and graph transformations that reduce a graph to a smaller one with equivalent reliability. Because of the complications involved in computing reliability, reliability optimization models rarely include explicit expressions for the reliability of the network. Instead, they often attempt to find the minimum-cost network design with some desired structural property, such as 2-connectivity (Monma and Shallcross 1989, Monma, Munson, and Pulleyblank 1990), k-connectivity (Bienstock, Brickell, and Monma 1990, Grötschel, Monma, and Stoer 1995), or special ring structures (Fortz and Labbé 2002).

The key difference between the network reliability models discussed so far and the models that we present in Chapters 5 and 6 is that these previous models are concerned entirely with connectivity. The only costs considered are those to construct the network, not the transportation cost after rerouting, which is the primary concern of our supply chain reliability models. The literature on power network reliability, however, often does consider the costs of power loss due to rerouting after a node or link failure. These models have the added complication that power cannot be routed along a single path but follows all paths in the network more or less uniformly (Hobbs et al. 2001).

2.3.2 Expected Covering Models

Several papers extend the classical maximum covering problem (Church and ReVelle 1974) to handle the randomness inherent in locating emergency services vehicles. The classical maximum covering problem assumes that a vehicle is always available when a call for service arrives, but this fails to model the congestion in such systems when multiple calls are received by a facility with limited resources. Daskin (1982) formulates the maximum expected covering location model (MEXCLM), which assumes a constant, system-wide probability that a server is busy when a call is received and seeks to maximize the total expected coverage; he solves the problem heuristically in Daskin (1983).

Other authors have criticized the assumption that the availability probability is uniform and have sought to improve on Daskin's model. ReVelle and Hogan (1989) present the maximum availability location problem (MALP), which allows the availability probability to vary among facility sites. They present a MIP formulation of the MALP whose LP relaxation frequently has integer optimal solutions, and they solve their model using a standard MIP solver. Ball and Lin (1993) justify the form of the coverage constraints in MEXCLM and MALP using reliability theory.

Larson (1974, 1975) introduced queuing-based location models which explicitly consider customers waiting for service in congested systems. His "hypercube model" is useful as a descriptive model, but because of its complexity, researchers have had difficulty incorporating it into optimization models. Berman, Larson, and Chiu (1985) incorporate the hypercube idea into a simple optimization model, presenting theoretical results about the trajectory of the optimal 1-median as the demand rate changes in a general network.

Daskin, Hogan, and ReVelle (1988) compare various stochastic covering problems in which the objective is to locate facilities to maximize expected coverage or the degree of backup coverage. Berman and Krass (2001) attempt to consolidate a wide range of approaches to facility location in congested systems, presenting a complex model that is illustrative but can be solved only for special cases.

2.3.3 Reliable Supply Chain Management

In the wake of the terrorist attacks on September 11, 2001, there has been a call for techniques for designing and operating supply chains that are resilient to disruptions of all sorts. Sheffi (2001), Simchi-Levi, Snyder, and Watson (2002), and Lynn (2002) make compelling arguments that supply chains are particularly vulnerable to intentional or accidental disruptions and suggest possible approaches for making them less so, but they do not present any rigorous models. We view the models presented in Chapters 5 and 6 as an outgrowth of this call for supply chain reliability models.

2.3.4 Other Related Research

Two other topics found in the literature are related to our reliability models. The first is the work on "a priori" optimization by Jaillet (1988, 1992) and Bertsimas, Jaillet, and Odoni (1990), whose goal is to find solutions to combinatorial optimization problems (e.g., the shortest path problem, minimum spanning tree problem, or traveling salesman problem) in which not all nodes may be present when the solution is implemented. For example, in the a priori traveling salesman problem, one wants a tour that is of minimum cost given a certain probability that each node will need to be visited; nodes that do not need to be visited are simply skipped. In general, the expected cost of a given solution can be computed efficiently, but the optimization problem is NP-hard, even when the underlying problem (e.g., the shortest path problem) is polynomially solvable.

The second related topic involves facility location problems in which each customer is assigned to multiple facilities, a strategy that we use in the models in Chapters 5 and 6. One such problem is the fault-tolerant facility location problem (Swamy and Shmoys 2003), a variant of the UFLP in which each customer i must be assigned to at least r_i facilities, where r_i is an input into the model. Most of the work on this problem is concerned with finding approximation algorithms for it. Fault-tolerant facility location problems are similar in spirit to ours since they require redundant backups to hedge against facility failures. However, these problems do not explicitly consider failures, and assignments are all given equal weight in the objective function. In our models, each customer receives a "primary" facility that serves it normally and one or more "backup" facilities that serve it when the primary facility fails. Our objective functions take this prioritization into account.

Another similar model is the vector assignment P-median problem (VAPMP; Weaver and Church 1983, Hooker and Garfinkel 1989), an extension of the PMP in which customers are served by multiple facilities based on preference and availability. For example, a given customer might receive 80% of its demand from its nearest facility, 15% from its second-nearest, and 5% from its third-nearest. These percentages are inputs to the model. In our reliability models in Chapters 5 and 6, the "higher-level" assignments are only used when the primary facilities fail; there are no pre-specified fractions of demand served by each facility.

2.4 Relaxation Methods for Facility Location Problems

In this section we review solution methods that have been proposed for facility location problems, focusing especially on Lagrangian relaxation methods for the capacitated fixed-charge location problem (CFLP). The goal is to familiarize the reader with some of the approaches that have been suggested for this problem since some of the models presented later in this dissertation entail similar challenges to those inherent in solving the CFLP. We first discuss the uncapacitated fixed-charge location problem (UFLP) and the P-median problem (PMP) and Lagrangian relaxation methods that have been proposed for them. We then examine the CFLP and its relaxations (Lagrangian and otherwise).

Lagrangian relaxation involves two nested optimization problems. For a given set of Lagrange multipliers, the inner optimization problem (the *subproblem*) provides a lower bound on the optimal objective value of the original problem (assuming this is a minimization problem). The outer optimization problem involves finding the best lower bound, taken over all possible Lagrange multipliers. The optimal objective value of the outer minimization problem is the *theoretical* lower bound provided by the Lagrangian relaxation method. Throughout this discussion, we are careful to draw a distinction between the theoretical lower bound and the *practical* lower bound—the best bound attained by a given implementation, which may fall short of the theoretical lower bound.

For any minimization problem, if z_{LR} is the theoretical bound from a Lagrangian relaxation and z_{LP} is the LP relaxation bound, then we have

$$z_{\mathrm{LR}} \geq z_{\mathrm{LP}}. \tag{2.14}$$

If the subproblem has the *integrality property* (i.e., it has an all-integer optimal solution even when the integrality constraints are relaxed), then the inequality (2.14) holds at equality—the Lagrangian bound is no better than the LP bound. On the other hand, if the subproblem does not have the integrality property, then the inequality in (2.14) is strict (Geoffrion 1974; Nemhauser and Wolsey 1988). Therefore it is desirable to develop

Lagrangian relaxations whose subproblems do not have the integrality property, provided that the subproblems can be solved quickly.

2.4.1 The PMP and UFLP

We define the following notation for the P-median problem:

Sets

I = set of customers, indexed by i

J = set of potential facility locations, indexed by j

Parameters

h_i = annual demand at customer $i \in I$

d_{ij} = cost per unit to ship from facility location $j \in J$ to customer $i \in I$

P = desired number of facilities to locate

Decision Variables

$$X_j = \begin{cases} 1, & \text{if a facility is established at location } j \in J \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, & \text{if a facility at location } j \in J \text{ serves customer } i \in I \\ 0, & \text{otherwise} \end{cases}$$

The PMP is formulated as follows:

$$\text{(PMP)} \quad \text{minimize} \quad \sum_{i\in I}\sum_{j\in J} h_i d_{ij} Y_{ij} \tag{2.15}$$

$$\text{subject to} \quad \sum_{j\in J} Y_{ij} = 1 \qquad \forall i \in I \tag{2.16}$$

$$Y_{ij} \leq X_j \qquad \forall i \in I, \forall j \in J \tag{2.17}$$

$$\sum_{j\in J} X_j = P \tag{2.18}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{2.19}$$

$$Y_{ij} \geq 0 \qquad \forall i \in I, \forall j \in J \tag{2.20}$$

The objective function (2.15) computes the total demand-weighted distance between customers and their assigned facilities. Constraints (2.16) require each customer to be assigned to a facility, and constraints (2.17) require that facility to be open. Constraint (2.18) requires P facilities to be opened. Constraints (2.19) and (2.20) require the location variables to be binary and the assignment variables to be non-negative.

The UFLP is formulated by replacing (2.15) with

$$\sum_{j\in J} f_j X_j + \sum_{i\in I}\sum_{j\in J} h_i d_{ij} Y_{ij} \tag{2.21}$$

and omitting constraint (2.18). In the UFLP, d_{ij} is generally taken to be a transportation cost rather than simply a distance. In either problem, the linking constraints (2.17) are sometimes replaced by

$$\sum_{i\in I} Y_{ij} \leq n X_j \qquad \forall j \in J, \tag{2.22}$$

but these constraints are known to provide a weaker LP relaxation than (2.17).

The most common Lagrangian relaxation algorithm for these problems is to relax the assignment constraints (2.16). This method was proposed for the UFLP by Geoffrion (1974) and for the PMP by Cornuejols, Fisher, and Nemhauser (1977). For the PMP, the Lagrangian subproblem is as follows:

$$\text{(PMP-LR)} \quad \underset{\lambda \geq 0}{\text{maximize}} \quad \underset{X,Y}{\text{minimize}} \quad \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} + \sum_{i \in I} \lambda_i \left(1 - \sum_{j \in J} Y_{ij} \right)$$

$$= \sum_{i \in I} \sum_{j \in J} (h_i d_{ij} - \lambda_i) Y_{ij} + \sum_{i \in I} \lambda_i \tag{2.23}$$

$$\text{subject to} \quad Y_{ij} \leq X_j \qquad \forall i \in I, \forall j \in I \tag{2.24}$$

$$\sum_{j \in J} X_j = P \tag{2.25}$$

$$X_j \in \{0, 1\} \qquad \forall j \in I \tag{2.26}$$

$$Y_{ij} \geq 0 \qquad \forall i \in I, \forall j \in I \tag{2.27}$$

We can restrict $\lambda \geq 0$ since if $\lambda_i < 0$, then $h_i d_{ij} - \lambda_i > 0$ and it is never advantageous to set $Y_{ij} = 1$ for any j; thus if $\lambda_i < 0$, a tighter bound can always be attained by setting $\lambda_i = 0$. To solve (PMP-LR) for a given λ, we compute the *benefit* (or contribution to the objective function) of opening each facility j:

$$\gamma_j = \sum_{i \in I} \min\{0, h_i d_{ij} - \lambda_i\}. \tag{2.28}$$

We then set $X_j = 1$ for the P facilities with the smallest γ_j and set $Y_{ij} = 1$ if $X_j = 1$ and $h_i d_{ij} - \lambda_i < 0$. To solve (PMP-LR), we must maximize over λ; this is done using subgradient optimization (see Fisher 1981, 1985 or Daskin 1995).

This procedure can be modified to solve the UFLP by adding $\sum_{j \in J} f_j X_j$ to the objective function, removing constraint (2.25), and setting $X_j = 1$ if $\gamma_j + f_j < 0$, or if

$\gamma_k + f_k \geq 0$ for all $k \in J$ but is smallest for j, since at least one facility must be open in any feasible solution. This method has been found to produce extremely tight bounds for both problems. This is because the Lagrangian bound is equal to the LP bound (since the Lagrangian subproblems have the integrality property), and both problems generally have very tight LP bounds. An analytical result is known about the bound for the PMP: Cornuejols et al. show that

$$\frac{Z_G - Z_{LR}}{Z_{LR}} \leq \left(\frac{P-1}{P}\right)^P < \frac{1}{e}, \tag{2.29}$$

where Z_{LR} is the Lagrangian bound and Z_G is the upper bound obtained from a particular greedy heuristic.

Christofides and Beasley (1982) compare two Lagrangian relaxations of the P-median problem, one in which the assignment constraints are relaxed (discussed above) and one in which the linking constraints are relaxed. When the linking constraints are relaxed, the subproblem decomposes into an X-problem and a Y-problem; both can be solved easily for given λ. They find empirically that the former relaxation results in a tighter bound (often 0%, but never more than 1% for their test problems) than the latter (which attained bounds between 0% and 7.4%). The reason for the difference lies in the fact that Christofides and Beasley use the "weak" linking constraints (2.22) instead of the "strong" form (2.17). The subproblem produced by relaxing the assignment constraints does not have the integrality property (for a given j, X_j will be set equal to $\sum_{i \in I} Y_{ij}/n$ if it is allowed to be fractional), whereas that produced by relaxing the linking constraints does. Since the former subproblem is solved to integer optimality, a tighter bound is attained. If the authors had used the strong linking constraints, the subproblems from both relaxations would have had the integrality property, and the two relaxations would have the same theoretical bound.

A different relaxation for the P-median problem was proposed by Hanjoul and Peeters (1985), who relax constraint (2.25). The resulting subproblem that is equivalent to the UFLP, which they solve using Erlenkotter's (1978) DUALOC algorithm. This subproblem is obviously harder than the subproblems that result when either (2.16) or (2.17) are relaxed, but it needs to be solved fewer times since there is only a single Lagrange multiplier to optimize over. The authors compare this relaxation to the (PMP-LR) and find the two to be roughly equivalent in terms of CPU time. They note that their relaxation provides tighter bounds since the subproblem does not have the integrality property, but they do not present any computational results to illustrate this claim. This method is similar to that of Mirchandani, Oudjit, and Wong (1985) for the stochastic PMP (see Section 2.2.1.2).

2.4.2 The CFLP: Notation and Formulation

We add following notation to that defined in Section 2.4.1:

Parameters

f_j = annual fixed cost to establish a facility at location $j \in J$

b_j = maximum annual capacity or throughput of a facility located at site $j \in J$

One formulation of the CFLP is as follows:

$$\text{(CFLP)} \qquad \text{minimize} \qquad \sum_{j \in J} f_j X_j + \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} \tag{2.30}$$

$$\text{subject to} \qquad \sum_{j \in J} Y_{ij} = 1 \qquad \forall i \in I \tag{D}$$

$$Y_{ij} \leq X_j \qquad \forall i \in I, \forall j \in J \tag{B}$$

$$\sum_{i \in I} h_i Y_{ij} \leq b_j X_j \qquad \forall j \in J \tag{C}$$

$$\sum_{j \in J} b_j X_j \geq \sum_{i \in I} h_i \tag{T}$$

$$X_j \in \{0, 1\} \qquad \forall j \in J \tag{I}$$

$$0 \leq X_j, Y_{ij} \leq 1 \qquad \forall i \in I, \forall j \in J \tag{N}$$

The letters labeling the constraints will be used to notate the various relaxations discussed below; this notation is taken from Cornuejols, Sridharan, and Thizy (1991), which we discuss in Section 2.4.3. The objective function (2.30) minimizes the sum of the fixed costs for locating facilities and the transportation costs. The **D**emand constraints (D) require each customer to be assigned to a facility. The variable upper-**B**ound constraints (B) require that facility to be open. Constraints (C) require the total volume assigned to facility j to be no more than its **C**apacity. Constraints (T) require the **T**otal capacity of the facilities opened to exceed the total demand; these constraints are redundant for the IP formulation but tighten some of the relaxations discussed below. Finally, constraints (I) and (N) require the location variables to be **I**nteger and all variables to be **N**on-negative.

Several variations of this model are possible. For example, some authors require the assignment variables to be binary, enforcing a "single-sourcing" constraint. (Because of the capacities, optimal solutions do not necessarily have integer Y variables, as they do in the UFLP.) In some formulations, constraints (B) or (T) are omitted; they are redundant in the IP formulation given above, but their inclusion makes for tighter LP or Lagrangian relaxations. Other formulations replace constraints (C) with

$$\sum_{i \in I} h_i Y_{ij} \leq b_j \qquad \forall j \in J. \tag{C$'$}$$

Let Z be the optimal IP objective value from (CFLP). Following Cornuejols et al., we will represent Lagrangian relaxations using subscripts and "complete" relaxations (i.e., omitting the constraints entirely) using superscripts. Thus, Z_D is the bound from relaxing constraints (D) using Lagrangian relaxation, Z^T is the bound from omitting the total capacity constraints (T), and Z_C^{BI} is the bound from omitting the linking constraints (B) and the integrality constraints (I) and relaxing the capacity constraints (C).

2.4.3 The CFLP: Relaxations

Davis and Ray (1969) solve the CFLP using branch-and-bound, solving the dual of the LP relaxation at each node using Dantzig–Wolfe decomposition, obtaining the bound Z^I, known as the "strong" LP relaxation of (CFLP). Akinc and Khumawala (1977) also propose an LP-relaxation/branch-and-bound method to solve the CFLP; they solve the "weak" LP relaxation in which (I) and (B) are omitted, but they tighten the formulation using ad-hoc rules.

By far, the most common method for solving the CFLP is Lagrangian relaxation. One of the first papers to propose such an algorithm is by Nauss (1978b). Nauss omits constraints (B) and relaxes the assignment constraints (D). The resulting subproblem reduces to a continuous knapsack problem (KP) for each j and a single 0–1 KP to decide which facilities to open, obeying constraint (T). The bound obtained from Nauss's relaxation is Z_D^B, though in his computational results, Nauss obtains a weaker bound because he only solves the continuous version of the 0–1 KP.

Christofides and Beasley's (1983) CFLP model is similar to Nauss's but is somewhat richer in that it includes minimum throughput constraints (as well as maximum throughput (C)); it also replaces (T) with upper and lower bounds on the number of facilities that may be opened. Like Nauss, Christofides and Beasley and relax (D), but their subproblem does not require a 0–1 KP because they omit constraints (T). We represent the bound from their relaxation by Z'_D. They also derive penalties for fixing variables to 0 or 1 after processing at the root node. Sridharan (1991) enhances Christofides and Beasley's model (minus the min-throughput constraints) by allowing upper-bound constraints on disjoint subsets of the location variables; these side constraints allow one to model multiple facility sizes at each location, at most one of which may be chosen. Their algorithm is similar to Christofides and Beasley's.

Klincewicz and Luss (1986) include integrality constraints for the Y variables and solve (CFLP) by relaxing the capacity constraints (C). The resulting subproblem is equivalent to the UFLP, and the authors solve it using Erlenkotter's (1978) DUALOC algorithm. They report LB–UB gaps of as high as 11%, though it is not clear whether the size of the gap is due more to poor lower or upper bounds. Since their subproblem does not have the integrality property (because it is equivalent to the UFLP, whose LP relaxation is not guaranteed to produce integer solutions), the lower bound from Klincewicz and Luss's relaxation (Z_C) is tighter than Z'_D, suggesting that Klincewicz and Luss's upper bound is loose, rather than their lower bound. On the other hand, Darby-Dowman and Lewis (1988) show that for a particular class of problems, Klincewicz and Luss's relaxation will always produce solutions that are capacity-infeasible, and that for these problems, the lower bound produced will not be particularly tight. Fortunately, this class of problems is somewhat limited: all problems in the class have the property that in their uncapacitated form, the optimal solution has only a single facility open.

Van Roy (1986) also relaxes (C), but instead of solving via straightforward Lagrangian relaxation, he presents a cross-decomposition algorithm for the CFLP. Cross-decomposition is a hybrid of Lagrangian relaxation and Benders decomposition. He shows in this paper and an earlier one (Van Roy 1983) that the Lagrangian and Benders subproblems are in a certain sense master problems for one another, and uses this result to construct an inner algorithm in which the two methods "ping pong" off one another; when this method stops making improvement, the algorithm reverts to an outer algorithm, which is either a Benders or Dantzig–Wolfe master problem that provides a new (primal or dual, respectively) variable to begin the inner algorithm again. His computational results are impressive, requiring few iterations of the outer algorithm and only a few seconds of CPU time for problems with up to 100 facilities and 200 customers.

Barcelo, Fernandez, and Jörnsten (1991) propose an algorithm for the CFLP based on variable-splitting (sometimes called Lagrangian decomposition). The idea is to introduce a new set of variables W to mirror the assignment variables Y. Each set of constraints is formulated using either Y or W to obtain a particular split. The W variables are forced equal to the Y variables by a new set of constraints:

$$\text{(CFLP-VS)} \qquad \text{minimize} \qquad \sum_{j\in J} f_j X_j + \beta \sum_{i\in I}\sum_{j\in J} h_i d_{ij} Y_{ij} + (1-\beta)\sum_{i\in I}\sum_{j\in J} h_i d_{ij} W_{ij} \tag{2.31}$$

$$\text{subject to} \qquad \sum_{j\in J} W_{ij} = 1 \qquad \forall i \in I \tag{DW}$$

$$\sum_{i\in I} h_i Y_{ij} \le b_j X_j \qquad \forall j \in J \tag{CXY}$$

$$\sum_{j\in J} b_j X_j \ge \sum_{i\in I} h_i \tag{TX}$$

$$W_{ij} = Y_{ij} \qquad \forall i \in I, \forall j \in J \tag{V}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{IX}$$

$$0 \le Y_{ij} \le 1 \qquad \forall i \in I, \forall j \in J \tag{NY}$$

$$0 \le W_{ij} \le 1 \qquad \forall i \in I, \forall j \in J \tag{NW}$$

where $0 \le \beta \le 1$ is a parameter. Only constraints (V) are relaxed using Lagrangian relaxation. The resulting subproblem decomposes into two problems, one involving only X and Y, which reduces to continuous knapsack problems for each j and a 0–1 KP to decide which facilities to open (as in Nauss 1978b), and one involving only W, which reduces to a trivial multiple-choice problem. Intuition suggests that by keeping all of the "interesting" constraints and relaxing only the new constraints, one obtains a bound at least as great as that from ordinary Lagrangian relaxation. This intuition is correct to a point, but not entirely, as discussed below.

Table 2.1: Relaxations for the CFLP.

Reference	Bound	Comments
Davis and Ray (1969)	Z^I	Solve dual of "strong" LP relaxation by Benders decomposition, then branch-and-bound
Akinc and Khumawala (1977)	Z^{BI}	Solve "weak" LP relaxation in branch-and-bound scheme with ad-hoc tightening rules
Nauss (1978)	Z_D^B	Subproblem = $\|J\|$ continuous KPs and one 0–1 KP
Christofides and Beasley (1983)	Z'_D	Add min throughput constraints, min and max cardinality constraints; remove (T); subproblem = $\|J\|$ continuous KPs
Klincewicz and Luss (1986)	Z_C	Single-sourcing; subproblem = UFLP
Van Roy (1986)	Z_C	Solves via cross-decomposition rather than Lagrangian relaxation
Sridharan (1991)	N/A	Includes side constraints to model multiple facility sizes; extension of Christofides and Beasley's algorithm
Barcelo, Fernandez, and Jörnsten (1991)	$Z_{D/CT}$	Variable-splitting algorithm

The relaxations discussed thus far in this section are summarized in Table 2.1.

Cornuejols, Sridharan, and Thizy (1991) provide dominance relationships among the theoretical bounds from the various relaxations of (CFLP). As noted above, these relaxations include both Lagrangian relaxations, in which constraints are dualized, and "complete" relaxations, in which constraints are omitted entirely. Their first result is that of the 41 possible relaxations of (CFLP), only 7 yield distinct bounds. The relaxations discussed thus far (except that of Sridharan (1991)) relate as follows:

$$Z^{BI} \le Z^I \le Z_C \le Z \tag{2.32a}$$

$$Z'_D \le Z_D^B = Z_D \le Z_C \tag{2.32b}$$

$$Z^{BI} \le Z_D^B = Z_D \tag{2.32c}$$

where Z is the optimal IP value. Moreover, each inequality in (2.32) is strict for some instances. (We have not discussed any papers considering Z_D; we include it here because it figures into the discussion of variable-splitting that follows.)

Cornuejols et al. also discuss bounds obtained from variable-splitting, with some surprising results. Let $Z_{D/CT}$ be the bound obtained by relaxing (V)—the notation indicates that the Demand constraints are in one set of the "split," the Capacity and Total capacity constraints are in the other. Then

$$Z_{D/CT} \geq Z_D \qquad \text{and} \qquad Z_{D/CT} \geq Z_{CT} \tag{2.33}$$

(see Guignard and Kim 1987), confirming the intuition that the variable-splitting bound is at least as tight as the corresponding simple Lagrangian bound. On the other hand, Cornuejols et al. show that the first inequality holds at equality, meaning variable-splitting does not offer any advantage over the bound Z_D. (The second inequality in (2.33) is strict for some instances.) Moreover, they show that

$$Z_{D/CT} \leq Z_C, \tag{2.34}$$

and that this inequality is strict for some instances. In fact, they find empirically that the relative error for Z_C (i.e., $(Z - Z_C)/Z$) is about half that for $Z_D = Z_{D/CT}$ for tightly constrained problems, and that the difference is even more pronounced for less capacitated problems. On the other hand, both bounds tend to be quite tight: $Z_C \leq 1\%$ and $Z_D \leq 3\%$ for all problems tested.

Geoffrion and McBride (1978) also offer a theoretical discussion of bounds for the CFLP. They omit constraints (B) and (T) but include minimum throughput constraints as well as any **A**dditional linear constraints on the X and Y variables, which we will denote (A). (A) may include, for example, cardinality constraints (open between 3 and 8 facilities), precedence constraints (don't open facility 4 if facility 2 is opened), and so on. They relax the demand constraints (D) and the additional constraints (A), attaining the bound Z_{DA}^{BT}. The subproblem reduces to a continuous KP for each facility j. Their main result is that

$$Z^{IBT} \le \hat{Z}_{DA}^{BT} \le Z_{DA}^{BT} = Z^{IT} \le Z, \tag{2.35}$$

where Z^{IBT} is the LP relaxation of their formulation, $\hat{Z}_{DA}^{BT}$ is the Lagrangian bound attained by relaxing (D) and (A) and setting the Lagrange multipliers equal to the corresponding optimal dual values from the LP relaxation, Z_{DA}^{BT} is the optimal Lagrangian bound (i.e., maximizing over the Lagrange multipliers), Z^{IT} is the LP bound when the linking constraints (B) are included in the formulation, and Z is the optimal IP value. The authors find empirically that the gap between Z^{IBT} and Z averages around 7%, and that about 70% of this gap is accounted for by the first inequality. The optimal Lagrange multipliers provide a tighter bound, which is equal to the "strong" LP relaxation bound Z^{IT}. (Above, we listed Z^I as the strong LP relaxation, but Cornuejols et al. show $Z^{IT} = Z^I$.) According to Geoffrion and McBride, the last inequality entails an average gap of only 0.6% or so.

Another interesting variation on Lagrangian relaxation that has been proposed for the CFLP is an algorithm proposed by Barahona and Chudak (1999a), which extends a similar algorithm (Barahona and Chudak 1999b) for the UFLP. Barahona and Chudak's algorithm is a heuristic that combines the volume algorithm and randomized rounding. The volume algorithm (Barahona and Anbil 2000) is essentially a Lagrangian method that gradually builds a solution that is close to feasible by taking convex combinations of the solutions found so far. Although each solution found by the Lagrangian procedure is binary, the "combined" solution will be fractional and will approximate the solution to the LP relaxation, based on a theorem in linear programming duality. The Lagrange multipliers are updated using an enhanced version of subgradient optimization. The idea behind randomized rounding (Raghavan and Thompson 1987) is to take the fractional solutions from the LP relaxation (or, in this case, from the approximate LP solution found using the volume algorithm) and round the facility location variable X_j to 1 with probability X_j and to 0 with probability $1 - X_j$. Once facilities have been opened by randomized rounding, assignments are made by solving a transportation problem. Computational results on problems with up to 1000 nodes show less than 1% relative error, but long run times.

Nozick (2001) considers a model that adds a coverage constraint of the form

$$\sum_{i \in I} \sum_{j \in J} h_i q_{ij} Y_{ij} \leq V \qquad (\text{C}'')$$

to the UFLP, where q_{ij} is 0 if facility j is within a given coverage distance of customer i and V is a desired bound on the total demand not served by a facility within the coverage distance. This constraint is like an aggregated form of (C′). She tests two Lagrangian relaxations, one in which (D) and (C″) are relaxed and one in which (B) and (C″) are relaxed (constraints (C) and (T) are omitted). She finds the latter relaxation to yield consistently tighter bounds. This is surprising since both relaxations have the integrality property and have the same theoretical bound; since the set (B) contains more constraints than (D), one would expect the subgradient optimization procedure to converge faster for the former relaxation.

Finally, we mention the informative article by Holmberg (1998), which discusses theoretical aspects of Lagrangian relaxation, Dantzig–Wolfe decomposition, Benders decomposition, cross decomposition, variable-splitting, and another technique called constraint duplication (essentially the dual of variable splitting), illustrating each with its application to the CFLP. The reader is referred to this article for more information about these techniques and how they relate to one another.

2.5 Location–Inventory Models

The location model with risk pooling (LMRP) presented in this section draws from classical inventory theory (see the general texts of Graves, Rinnooy Kan, and Zipkin (1993), Nahmias (2001), or Zipkin (1997)). In particular, it draws from the seminal work by Eppen (1979) on risk pooling. Eppen showed that if demands are normally distributed and uncorrelated, the cost of a newsboy-type inventory system increases with the square root of the number of DCs. The LMRP itself was first developed by Shen (2000) and Shen, Coullard, and Daskin (2003); both references present a column generation algorithm for solving the LMRP. Daskin, Coullard, and Shen (2002) present a Lagrangian-relaxation–based algorithm for the same problem. The algorithms in both papers make the simplifying assumption that the variance-to-mean ratio is the same for all retailers' demands. This assumption makes the subproblems easy to solve. Without this assumption, the problem can still be solved, but Shen et al.'s algorithm for the subproblem in this case runs in $O(n^7 \log n)$ time, where n is the number of retailers. A faster, $O(n^2 \log n)$, algorithm is presented by Shu, Teo, and Shen (2001).

A handful of other location–inventory models have appeared in the literature. Barahona and Jensen (1998) use Dantzig–Wolfe decomposition coupled with subgradient optimization to solve a location problem with a fixed inventory cost for stocking a given product at a given DC. Their model is tractable but not very rich. Erlebacher and Meller (2000) use various heuristic techniques to solve a joint location–inventory problem with a highly non-linear objective function, with limited success. Teo, Ou, and Goh (2001) present a $\sqrt{2}$-approximation algorithm for the problem of choosing DCs to minimize location and inventory costs, ignoring transportation costs.

Nozick and Turnquist (2001b) present a model to choose DC locations, allocations, and stocking policies in a multi-product system. Their model can be used, for example, to decide which products to stock at a central plant, which to stock at regional DCs, and which not to stock at all (i.e., produce in a make-to-order fashion). They propose an iterative approach that alternately solves a UFLP (with inventory accounted for by a linear approximation, justified by Nozick and Turnquist 1998) and a stocking problem (for a fixed set of DC locations); both problems are solved heuristically. Nozick and Turnquist (2001a) consider a multi-objective model that embeds inventory cost and coverage into the UFLP, again linearizing the inventory cost. These models are similar in spirit to the LMRP, but they do not handle risk-pooling since inventory costs are linearized (removing the concavity necessary for risk-pooling to be effective) and DC–retailer allocations are made based only on distance, not inventory.

In the remainder of this section we describe the LMRP.

2.5.1 LMRP: Problem Statement

Shen, Coullard, and Daskin ("SCD"; 2003) and Daskin, Coullard, and Shen ("DCS"; 2002) formulate a location model with risk pooling, which we will refer to as the LMRP. Given a set of retailers, the problem is to choose a subset of the retailers to serve as distribution centers (DCs) for the other retailers.[2] (We will use the terms "DC" and "facility" interchangeably.) These DCs will order a single product from a single supplier at regular intervals and distribute the product to the retailers. The DCs will hold *working inventory* representing product that has been ordered from the supplier but not yet requested by the retailers and *safety stock inventory* designed to buffer the system against stockouts during ordering lead times, which are fixed and deterministic.

Let I be the set of retailers, which face independent normal random demands. The firm pays a fixed location cost for establishing a DC at a retailer, as well as a fixed cost for each order placed at a DC and a holding cost for inventory. There are fixed and variable costs for shipping from the supplier to DCs and a variable cost for shipping from DCs to retailers. We wish to choose DC locations to minimize the sum of all of these costs. The notation is as follows:

[2] The set of potential DC locations need not be the same as the set of retailers, but throughout our discussion of the LMRP and its stochastic extensions, we will assume WLOG that they are equal. If there are retailers that are not potential DC sites, their fixed location costs can be set to ∞, and if there are DC sites that are not retailers, their demand can be set to 0.

Parameters

Demand

μ_i = mean daily demand at retailer i, for $i \in I$

σ_i^2 = variance of daily demand at retailer i, for $i \in I$

Costs

d_{ij} = per-unit cost to ship from a DC located at retailer j to retailer i, for $i, j \in I$

f_j = fixed cost per year of locating a DC at retailer j, for $j \in I$

F_j = fixed cost per order placed to the supplier by a DC located at retailer j, for $j \in I$

g_j = fixed cost per shipment from the supplier to a DC located at retailer j, for $j \in I$

a_j = per-unit cost to ship from the supplier to a DC located at retailer j, for $j \in I$

h = inventory holding cost per unit per year

Weights

β = weight factor associated with transportation cost, $\beta \geq 0$

θ = weight factor associated with inventory cost, $\theta \geq 0$

Other Parameters

α = desired percentage of retailer lead-time orders satisfied

z_α = standard normal deviate such that $P(z \leq z_\alpha) = \alpha$

L_j = lead time in days for orders placed by a DC located at retailer j, for $j \in J$

χ = number of days per year

In the notation above (and the analysis below), the time horizon of the model is assumed to be one year. However, one could easily choose a different time horizon, adjusting the values of f_j, h, and χ accordingly.

Suppose for the moment that we know which retailers are assigned to facility j. Let $\mathbf{S}$ be the set of retailers assigned to facility j, and let D be the expected annual demand of retailers in $\mathbf{S}$: $D = \chi \sum_{i \in \mathbf{S}} \mu_i$. Let n be the (as yet undetermined) number of orders that DC j places each year; then the expected size of a shipment is D/n. Using the notation above, we can compute the annual cost of ordering, transporting, and holding inventory at distribution center j:

$$F_j n + \beta \left(g_j + \frac{a_j D}{n} \right) n + \theta \frac{hD}{2n}. \tag{2.36}$$

The first term represents the fixed cost of placing n orders; the second represents the cost of transporting n orders of size D/n; and the third term represents the cost of holding an average of $D/2n$ units of inventory per year. To determine the optimal number of orders per year, n^*, we take the derivative of (2.36) with respect to n:

$$F_j + \beta g_j - \theta \frac{hD}{2n^2}. \tag{2.37}$$

Setting (2.37) equal to 0 and solving for n, we get $n^* = \sqrt{\frac{\theta h D}{2(F_j + \beta g_j)}}$. Plugging n^* into (2.36), we get a total cost of

$$F_j \sqrt{\frac{\theta h D}{2(F_j + \beta g_j)}} + \beta \left(g_j \sqrt{\frac{\theta h D}{2(F_j + \beta g_j)}} + a_j D \right) + \theta \frac{hD}{2} \sqrt{\frac{2(F_j + \beta g_j)}{\theta h D}}$$

$$= \sqrt{2\theta h D(F_j + \beta g_j)} + \beta a_j D. \tag{2.38}$$

This expression represents the total cost of obtaining working inventory at facility j, assuming j follows the optimal EOQ-style inventory ordering policy. To compute the total expected annual cost, we still need to add the fixed cost of opening facility j, the cost of transporting product from the DC to the retailers, and the safety stock cost at j. The fixed cost is simply equal to f_j. The DC–retailer transportation cost is given by

$$\beta\chi \sum_{i \in \mathbf{S}} d_{ij}\mu_i.$$

To compute the safety stock cost, note that the variance of demand during the lead time is $L_j \sum_{i \in \mathbf{S}} \sigma_i^2$ since the retailers' demands are independent. The amount of safety stock needed to ensure that stockouts occur during the lead time with a probability no greater than α is

$$z_\alpha \sqrt{L_j \sum_{i \in \mathbf{S}} \sigma_i^2},$$

and the cost of holding this much safety stock is

$$\theta h z_\alpha \sqrt{L_j \sum_{i \in \mathbf{S}} \sigma_i^2}.$$

Note that we do not need to figure the safety stock in the derivation of the working inventory cost (2.38) because the level of safety stock has no impact on the size or frequency of orders placed by facility j.

Combining all costs, we get a total expected annual cost of operating facility j:

$$f_j + \beta\chi \sum_{i \in \mathbf{S}} d_{ij}\mu_i + \sqrt{2\theta h D(F_j + \beta g_j)} + \beta a_j D + \theta h z_\alpha \sqrt{L_j \sum_{i \in \mathbf{S}} \sigma_i^2}. \tag{2.39}$$

But this cost assumes that we know the retailers assigned to facility j. Instead, we want to determine the set $\mathbf{S}$ within the model and compute the expected cost "on the fly."

To this end, we define the following variables:

Decision Variables

$$X_j = \begin{cases} 1, & \text{if we locate a DC at retailer } j \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, & \text{if retailer } i \text{ is served by a DC at retailer } j \\ 0, & \text{otherwise} \end{cases}$$

We can now formulate the location model with risk pooling (LMRP):

$$\begin{aligned}
\text{(LMRP)} \quad \text{minimize} \quad & \sum_{j\in I} f_j X_j + \beta\chi \sum_{j\in I}\sum_{i\in I} d_{ij}\mu_i Y_{ij} \\
& + \sum_{j\in I}\sqrt{2\theta h\chi(F_j+\beta g_j)\sum_{i\in I}\mu_i Y_{ij}} + \beta\chi\sum_{j\in I}\sum_{i\in I} a_j\mu_i Y_{ij} \\
& + \theta h z_\alpha \sum_{j\in I}\sqrt{\sum_{i\in I} L_j\sigma_i^2 Y_{ij}} \\
= & \sum_{j\in I} f_j X_j + \beta\chi\sum_{j\in I}\sum_{i\in I}\mu_i(d_{ij}+a_j)Y_{ij} \\
& + \sum_{j\in I}\sqrt{2\theta h\chi(F_j+\beta g_j)\sum_{i\in I}\mu_i Y_{ij}} + \theta h z_\alpha\sum_{j\in I}\sqrt{\sum_{i\in I} L_j\sigma_i^2 Y_{ij}} \\
= & \sum_{j\in I}\left\{ f_j X_j + \sum_{i\in I}\hat{d}_{ij}Y_{ij} + K_j\sqrt{\sum_{i\in I}\mu_i Y_{ij}} + \Theta\sqrt{\sum_{i\in I} L_j\sigma_i^2 Y_{ij}} \right\} \qquad (2.40)
\end{aligned}$$

$$\text{subject to} \quad \sum_{j \in I} Y_{ij} = 1 \qquad \forall i \in I \tag{2.41}$$

$$Y_{ij} \leq X_j \qquad \forall i \in I, \forall j \in I \tag{2.42}$$

$$X_j \in \{0,1\} \qquad \forall j \in I \tag{2.43}$$

$$Y_{ij} \in \{0,1\} \qquad \forall i \in I, \forall j \in I \tag{2.44}$$

The notation in the last line of the objective function is as follows:

$$\begin{aligned} \hat{d}_{ij} &= \beta\chi\mu_i(d_{ij} + a_j) \\ K_j &= \sqrt{2\theta h\chi(F_j + \beta g_j)} \\ \Theta &= \theta h z_\alpha \end{aligned}$$

The objective function (2.40) sums the fixed cost of locating facilities, the DC–retailer transportation cost, the working inventory cost (which includes ordering, supplier–DC transportation, and holding costs), and the safety stock cost. Constraints (2.41) require each retailer be assigned to exactly one DC. Constraints (2.42) prohibit a retailer from being assigned to a DC that has not been opened. Constraints (2.43) and (2.44) are standard integrality constraints.

Note that if $\theta = 0$, problem (LMRP) is identical in form to the classical UFLP. Therefore, the LMRP is NP-hard. Unfortunately, the square-root terms in the objective function make the standard algorithms for the UFLP inapplicable to the problem when $\theta > 0$. However, SCD and DCS both use modifications of standard algorithms to solve this problem. Their algorithms depend on the following assumption:

Assumption 2.1 *The variance-to-mean ratio* σ_i^2/μ_i *is identical for all retailers. That* *is, for all* $i \in I$, $\sigma_i^2/\mu_i = \gamma$ *for some constant* $\gamma \geq 0$.[3]

This assumption allows us to further simplify the objective function

$$\begin{aligned} & \sum_{j \in I} \left\{ f_j X_j + \sum_{i \in I} \hat{d}_{ij} Y_{ij} + K_j \sqrt{\sum_{i \in I} \mu_i Y_{ij}} + \Theta \sqrt{\sum_{i \in I} L_j \sigma_i^2 Y_{ij}} \right\} \\ = & \sum_{j \in I} \left\{ f_j X_j + \sum_{i \in I} \hat{d}_{ij} Y_{ij} + K_j \sqrt{\sum_{i \in I} \mu_i Y_{ij}} + \Theta \sqrt{\sum_{i \in I} L_j \gamma \mu_i Y_{ij}} \right\} \\ = & \sum_{j \in I} \left\{ f_j X_j + \sum_{i \in I} \hat{d}_{ij} Y_{ij} + \hat{K}_j \sqrt{\sum_{i \in I} \mu_i Y_{ij}} \right\} \end{aligned} \tag{2.45}$$

where

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

This revised objective function, with one square-root term instead of two, will make the solution procedure more efficient, though Shu, Teo, and Shen (2001) later reduced the complexity of the algorithm for the two-square-root objective function. If demands are Poisson, Assumption 2.1 is satisfied exactly. If not, the assumption may still be satisfied approximately. Another effect of Assumption 2.1 is that the optimal solution will never open retailer j as a DC but serve demands at j from a different DC, an odd circumstance that can happen if Assumption 2.1 is not satisfied. (See SCD for an example and DCS for a proof that the situation cannot arise if Assumption 2.1 holds.) Even with Assumption 2.1, however, it is possible that a retailer is served from a facility other than its closest. For the remainder of this section, we will assume that Assumption 2.1 holds.

[3] Or $\mu_i = \sigma_i^2 = 0$. This is useful for implementing the modeling trick described in footnote 2.

2.5.2 LMRP: Solution Procedure

SCD use a set-covering/column-generation approach to solve (LMRP), which we omit here. DCS propose a Lagrangian-relaxation–based algorithm for solving (LMRP). This algorithm serves as the basis for algorithms developed later in this dissertation. The algorithm is an extension of the standard Lagrangian relaxation algorithm for the UFLP (Geoffrion 1974).

2.5.2.1 Lower Bound

If we relax constraints (2.41), we obtain the following problem:

$$
\begin{aligned}
\text{(LMRP-LR)} \qquad \underset{\lambda \geq 0}{\text{maximize}} \ \mathcal{L}_\lambda = \underset{X,Y}{\text{minimize}} \qquad & \sum_{j \in I} \left\{ f_j X_j + \sum_{i \in I} \hat{d}_{ij} Y_{ij} + \hat{K}_j \sqrt{\sum_{i \in I} \mu_i Y_{ij}} \right\} \\
& + \sum_{i \in I} \lambda_i \left(1 - \sum_{j \in I} Y_{ij} \right) \\
= & \sum_{j \in I} \left\{ f_j X_j + \sum_{i \in I} (\hat{d}_{ij} - \lambda_i) Y_{ij} + \hat{K}_j \sqrt{\sum_{i \in I} \mu_i Y_{ij}} \right\} \\
& + \sum_{i \in I} \lambda_i \qquad (2.46)
\end{aligned}
$$

$$
\begin{aligned}
\text{subject to} \qquad & Y_{ij} \leq X_j \qquad && \forall i \in I, \forall j \in I \qquad (2.47) \\
& X_j \in \{0,1\} && \forall j \in I \qquad (2.48) \\
& Y_{ij} \in \{0,1\} && \forall i \in I, \forall j \in I \qquad (2.49)
\end{aligned}
$$

For fixed Lagrange multipliers λ, we can solve this problem by determining, for each $j \in I$, the benefit of opening a facility at retailer j. This benefit can be computed by solving the following subproblem:

$$(\text{SP}_j) \qquad \text{minimize} \quad \tilde{V}_j = \quad \sum_{i \in I} b_i Z_i + \sqrt{\sum_{i \in I} c_i Z_i} \tag{2.50}$$

$$\text{subject to} \qquad Z_i \in \{0,1\} \qquad \forall i \in I \tag{2.51}$$

where

$$b_i = \hat{d}_{ij} - \lambda_i$$

$$c_i = \hat{K}_j^2 \mu_i$$

$$Z_i = Y_{ij}.$$

Note that $c_i \geq 0$ for all i. This subproblem, which also arises as the pricing problem for the column-generation algorithm in SCD, is a non-linear integer program, but SCD show that it can be solved using the following $O(|I| \log |I|)$ algorithm:

Algorithm 2.1 (PRICING)

Step 1: Partition I into three sets as follows:

$$I^+ \quad = \quad \{i | b_i \geq 0\}$$

$$I^0 \quad = \quad \{i | b_i < 0 \text{ and } c_i = 0\}$$

$$I^- \quad = \quad \{i | b_i < 0 \text{ and } c_i > 0\}$$

Step 2: Sort the elements of I^- such that

$$\frac{b_1}{c_1} \leq \frac{b_2}{c_2} \leq \cdots \leq \frac{b_n}{c_n},$$

where $n = |I^-|$.

Step 3: Compute the partial sums

$$S_m = \sum_{i \in I^0} b_i + \sqrt{\sum_{i \in I^0} c_i} + \sum_{\substack{i=1 \\ i \in I^-}}^{m} b_i + \sqrt{\sum_{\substack{i=1 \\ i \in I^-}}^{m} c_i}$$

for $m = 0, \ldots, n$. (Note that the first square-root term will equal 0 by the definition of I^0.)

Step 4: Select the value of m that results in the minimum value of S_m, and set

$$Z_i = \begin{cases} 1, & \text{if } i \in I^0 \\ 1, & \text{if } i \in I^- \text{ and } i \leq m \\ 0, & \text{otherwise} \end{cases}$$

For each retailer j, the benefit of opening facility j is given by $\tilde{V}_j$, the optimal solution to (SP$_j$). The location variable X_j is set to 1 if

$$\tilde{V}_j + f_j < 0,$$

that is, if the net benefit after adding the fixed cost is still negative. (If no facility has a negative net benefit, we set $X_j = 1$ for the facility with the smallest net benefit.) The assignment variable Y_{ij} is set to 1 if $X_j = 1$ and $Z_i = 1$ in the optimal solution to (SP$_j$). The Lagrange multipliers are updated using subgradient optimization (see Fisher 1981, 1985 or Daskin 1995). The best value of $\mathcal{L}_\lambda$ found during the Lagrangian process serves as a lower bound on (2.40).

2.5.2.2 Upper Bound

Each time we solve (LMRP-LR), the current solution $(\hat{X}, \hat{Y})$ is used to obtain a feasible solution to (LMRP) greedily. If the cost of the resulting solution is less than the best upper bound found so far, DCS apply two improvement heuristics to it. In brief, the first heuristic involves re-assigning retailers from their currently assigned facility to a different one if doing so reduces the total cost. If at some point all of the demand assigned to a facility has been removed from the facility, one saves the fixed cost associated with the facility in addition to the other costs. The second procedure involves swapping a facility out of the solution in favor of a facility not currently in the solution, if doing so reduces the total cost; this procedure is similar to Teitz and Bart's (1968) procedure for the P-median problem.

2.5.2.3 Branch and Bound

If, when the Lagrangian procedure terminates, the best lower bound found is equal to the best upper bound (to within some pre-specified tolerance), we have found an optimal solution to (LMRP). Otherwise, a branch-and-bound procedure is employed to close the gap, with branching performed on the X_j (location) variables. At each node of the branch-and-bound tree, the facility selected for branching is the unfixed facility with the greatest assigned demand, or, if all facilities in the solution have already been forced open, we branch on an arbitrarily selected unforced facility. The variable is first forced to 0 and then to 1. Branching is done in a depth-first manner. The tree is fathomed at a given node if the lower bound at that node is greater than or equal to the objective value of the best feasible solution found anywhere in the tree to date, or if all facilities have been forced open or closed. A simple variable-fixing test is performed after processing at the root node to force some variables to 0 or 1 before any branching is performed. This test is very effective and takes negligible CPU time. It is described in greater detail for the SLMRP in Section 3.2.4.

2.5.2.4 Tightness of the Bound

As mentioned in the previous section, the lower bound from (LMRP-LR) tends to be extremely tight. To see why this might be so, first imagine that $\theta = 0$, that is, that we are dealing with a pure UFLP. Many location problems, including the UFLP, have a very tight LP-relaxation bound (ReVelle and Swain 1970, Schrage 1975), at least when the distance matrix satisfies the triangle inequality. Furthermore, it is well known that the theoretical Lagrangian bound is not less than the LP bound (see Nemhauser and Wolsey 1988 or Fisher 1981). Therefore the Lagrangian bound should also be very tight for the UFLP, and this property seems to hold for the non-linear LMRP, even when θ is large relative to β (and hence the objective function has a high degree of non-linearity).

2.6 Chapter Summary

In this chapter we reviewed the literature on robust optimization, focusing especially on location problems. We used "robust" to refer to problems that optimize based on uncertain future conditions, rather than the more narrow definition that uses robust to refer only to problems in which no probabilistic information is known. We discussed several different measures of robustness and presented examples of each from the literature. Of particular interest among the robustness measures is that of p-robustness, which we will return to in Chapter 4.

Although no models for reliable supply chain design (in the sense in which we define it) have appeared in the literature, we discussed several streams of research that relate to the reliability models we present in Chapters 5 and 6, including network reliability models and expected covering models. We also discussed relaxation methods for facility location problems, especially Lagrangian relaxation. Finally, we discussed location–inventory models, especially the location model with risk pooling (LMRP), which will serve as the basis for models formulated later in this dissertation.

Chapter 3

The Stochastic Location Model with Risk Pooling (SLMRP)

The LMRP model discussed in the previous chapter involves random demands, but estimates of demand means and variances (μ_i and σ_i^2) may be inaccurate due to poor forecasts, measurement errors, or changing demand patterns. In this chapter we present a model that allows the modeler to specify several possible future states, or scenarios. Each scenario dictates the demand and cost information that drives the supply chain model. This allows us to hedge against forecast errors or changes in parameters over time.

As we discussed in Chapter 1, uncertain parameters are modeled using either discrete scenarios or continuous distributions. We chose the scenario approach for a number of reasons. The primary reason is that it allows us to model dependence among random parameters. Future demands are likely to be correlated, as are costs. Under the continuous approach, such correlation could be modeled, but in all likelihood the problem would be intractable. Even without dependence, stochastic models with continuous parameters are extremely difficult to solve, and stochastic programming researchers have generally stayed away from them. Computational tractability is our second reason for using scenarios, as the solution techniques previously published for the LMRP can be extended to handle the scenario problem. Finally, as we discuss in Section 3.3, the scenario framework can be interpreted in a number of different ways, allowing us to use it to model and solve multi-commodity and multi-period versions of the LMRP.

3.1 Formulation

Suppose that demand means and variances, distances, and costs are random and are described by scenarios, each with a specified probability of occurrence. Location decisions (X) are scenario-independent: they must be made before it is known which scenario will be realized. Assignment decisions (Y) are scenario-*dependent*, so Y_{ij} becomes Y_{ijs}. Inventory decisions are also scenario-dependent, in that the levels of cycle and safety stock change once assignments are made and demand means and variances are known, though there are no explicit inventory variables. Note that there are now two levels of randomness: scenarios determine the means and variances of the demands, but once the scenario has been realized, demands are still random according to the specified probability distribution. Our goal is to choose facility locations to *minimize the expected cost of the system.*

Let S be the set of scenarios, indexed by s. We modify the notation from Section 2.5.1 as follows:

Parameters

Demand

μ_{is} = mean daily demand at retailer i in scenario s, for $i \in I,\ s \in S$

σ^2_{is} = variance of daily demand at retailer i in scenario s, for $i \in I,\ s \in S$

Costs

d_{ijs} = per-unit cost to ship from a DC located at retailer j to retailer i in scenario s, for $i, j \in I,\ s \in S$

Probabilities

q_s = probability that scenario s occurs, for $s \in S$

Variables

Assignment Variables

$$Y_{ijs} = \begin{cases} 1, & \text{if retailer } i \text{ is served by a DC at retailer } j \text{ in scenario } s \\ 0, & \text{otherwise} \end{cases}$$

In fact, any of the costs ($a_j,\ g_j$, etc.) other than f_j and any of the other parameters ($L_j,\ \chi$, etc.) can be scenario-dependent ($a_{js},\ g_{js},\ L_{js},\ \chi_s$, etc.); the analysis to follow can be modified in a straightforward way to incorporate these costs. For simplicity, however, we will assume that only demand means and variances and DC–retailer transportation costs are scenario-dependent.

We can now formulate the stochastic location model with risk pooling (SLMRP):

$$\text{(SLMRP)} \quad \text{minimize} \quad \sum_{s\in S}\sum_{j\in I} q_s \Bigg\{ f_j X_j + \beta\chi \sum_{i\in I} \mu_{is}(d_{ijs} + a_j) Y_{ijs} + \sqrt{2\theta h\chi(F_j + \beta g_j) \sum_{i\in I} \mu_{is} Y_{ijs}} + \theta h z_\alpha \sqrt{\sum_{i\in I} L_j \sigma_{is}^2 Y_{ijs}} \Bigg\}$$

$$= \sum_{s\in S}\sum_{j\in I} q_s \Bigg\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + K_j \sqrt{\sum_{i\in I} \mu_{is} Y_{ijs}} + \Theta \sqrt{\sum_{i\in I} L_j \sigma_{is}^2 Y_{ijs}} \Bigg\} \tag{3.1}$$

$$\text{subject to} \quad \sum_{j\in I} Y_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{3.2}$$

$$Y_{ijs} \le X_j \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{3.3}$$

$$X_j \in \{0,1\} \qquad \forall j \in I \tag{3.4}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{3.5}$$

The objective function (3.1) computes the expected value of the individual-scenario costs given in (2.40), with subscripts s added to the appropriate parameters and variables. In the last line of the objective function, one additional piece of notation is used:

$$\hat{d}_{ijs} = \beta\chi(d_{ijs} + a_j)\mu_{is}.$$

Constraints (3.2) require each retailer to be assigned to exactly one DC in each scenario. Constraints (3.3) prohibit a retailer from being assigned to a given DC in any scenario unless that DC has been opened. Constraints (3.4) and (3.5) are standard integrality constraints. Since the SLMRP reduces to the LMRP when $|S| = 1$, the SLMRP is NP-hard.

We will make the following assumption, which is the stochastic version of Assumption 2.1:

Assumption 3.1 *In each scenario $s \in S$, the variance-to-mean ratio σ_{is}^2/μ_{is} is identical for all retailers. That is, for each $s \in S$, there exists $\gamma_s \geq 0$ such that $\sigma_{is}^2/\mu_{is} = \gamma_s$ for all $i \in I$.*

Note that the variance-to-mean ratio γ_s may differ from scenario to scenario. This assumption allows us to rewrite the objective function (3.1) as follows:

$$\sum_{s\in S}\sum_{j\in I} q_s \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + K_j \sqrt{\sum_{i\in I} \mu_{is} Y_{ijs}} + \Theta \sqrt{\sum_{i\in I} L_j \sigma_{is}^2 Y_{ijs}} \right\}$$

$$= \sum_{s\in S}\sum_{j\in I} q_s \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + K_j \sqrt{\sum_{i\in I} \mu_{is} Y_{ijs}} + \Theta \sqrt{\sum_{i\in I} L_j \gamma_s \mu_{is} Y_{ijs}} \right\}$$

$$= \sum_{s\in S}\sum_{j\in I} q_s \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + \hat{K}_{js} \sqrt{\sum_{i\in I} \mu_{is} Y_{ijs}} \right\} \tag{3.6}$$

where

$$\hat{K}_{js} = K_j + \Theta\sqrt{L_j \gamma_s}.$$

Problem (SLMRP) looks like (LMRP) with $|I||S|$ retailers instead of $|I|$ and some of the parameters multiplied by the constant q_s. We will utilize this structure in our solution procedure.

3.2 Solution Procedure

As in the algorithm for (LMRP), we will relax the assignment constraints and use Lagrangian relaxation to solve (SLMRP).

3.2.1 Lower Bound

Relaxing constraints (3.2) with Lagrange multipliers λ_{is}, we get the following Lagrangian problem:

$$
\begin{aligned}
\text{(SLR)} \quad \underset{\lambda \geq 0}{\text{maximize}}\ \mathcal{L}_\lambda = \underset{X,Y}{\text{minimize}} \quad & \sum_{s\in S}\sum_{j\in I} q_s \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + \hat{K}_{js} \sqrt{\sum_{i\in I} \mu_{is} Y_{ijs}} \right\} \\
& + \sum_{s\in S}\sum_{i\in I} \lambda_{is} \left(1 - \sum_{j\in I} Y_{ijs} \right) \\
= & \sum_{s\in S}\sum_{j\in I} \left\{ q_s f_j X_j + \sum_{i\in I} (q_s \hat{d}_{ijs} - \lambda_{is}) Y_{ijs} \right. \\
& \left. + q_s \hat{K}_{js} \sqrt{\sum_{i\in I} \mu_{is} Y_{ijs}} \right\} + \sum_{s\in S}\sum_{i\in I} \lambda_{is} \qquad (3.7)
\end{aligned}
$$

$$
\begin{aligned}
\text{subject to} \quad & Y_{ijs} \leq X_j && \forall i \in I, \forall j \in I, \forall s \in S && (3.8) \\
& X_j \in \{0,1\} && \forall j \in I && (3.9) \\
& Y_{ijs} \in \{0,1\} && \forall i \in I, \forall j \in I, \forall s \in S && (3.10)
\end{aligned}
$$

We can restrict $\lambda \geq 0$ since if $\lambda_{is} < 0$, then $q_s \hat{d}_{ijs} - \lambda_{is} > 0$ and it is never advantageous to set $Y_{ijs} = 1$ for any j; if we set $\lambda_{is} = 0$, the last term of the objective function increases without affecting any of the other terms, thus tightening the Lagrangian bound.

For fixed values of λ, this problem decomposes by j and s. Ignoring for now the fixed location costs f_j, we can compute the *benefit* of using facility j in scenario s by solving

$$
\begin{aligned}
(\text{SSP}_{js}) \quad \text{minimize} \quad \tilde{V}_{js} \quad &= \quad \sum_{i\in I} b_i Z_i + \sqrt{\sum_{i\in I} c_i Z_i} && (3.11) \\
\text{subject to} \quad & Z_i \in \{0,1\} \qquad \forall i \in I && (3.12)
\end{aligned}
$$

where

$$b_i = q_s \hat{d}_{ijs} - \lambda_{is}$$

$$c_i = q_s^2 \hat{K}_{js}^2 \mu_{is}$$

$$Z_i = Y_{ijs}$$

This problem is equivalent to (SP$_j$), and for given j, s, it can be solved using Shen's algorithm, described in Section 2.5.2.

Now, if facility j is opened in one scenario, it must be opened in every scenario. Therefore the overall benefit of opening facility j is equal to the sum of all of the scenario-specific benefits:

$$\tilde{V}_j = \sum_{s \in S} \tilde{V}_{js}.$$

To solve (SLR) for fixed λ, we compute $\tilde{V}_{js}$ for each $j \in I, s \in S$, then compute $\tilde{V}_j$ for each j. The fixed cost to open facility j is equal to $f_j \sum_{s \in S} q_s = f_j$ since the scenario probabilities sum to 1. Therefore, we set $X_j = 1$ if

$$\tilde{V}_j + f_j < 0. \tag{3.13}$$

If $\tilde{V}_j + f_j \geq 0$ for all $j \in I$, then we set $X_j = 1$ for the j that minimizes $\tilde{V}_j + f_j$ since at least one facility must be open in any feasible solution. We set $Y_{ijs} = 1$ if $X_j = 1$ and $Z_i = 1$ in the optimal solution to (SSP$_{js}$). To solve the overall problem (SLR), we find the optimal values of λ using subgradient optimization (Fisher 1981, 1985; Daskin 1995). The best value of $\mathcal{L}_\lambda$ found during the Lagrangian process serves as a lower bound on (3.1).

Shen's algorithm for (SSP_{js}) has complexity $O(n \log n)$, where $n = |I|$. (If Assumption 3.1 does not hold, (SSP_{js}) has an additional square-root term in the objective function and can be solved in $O(n^2 \log n)$ time (Shu, Teo, and Shen 2001).) At each iteration of the Lagrangian procedure, we must solve this problem for all $j \in I$, $s \in S$, so the total complexity required to solve (SLR) for a given λ is $O(n^2|S| \log n)$.

3.2.2 Upper Bound

Each time we solve (SLR), we use the current solution $(\hat{X}, \hat{Y})$ to obtain a feasible solution to (SLMRP). For each $j \in I$, we open a DC at j if $\hat{X}_j = 1$ in the optimal solution to (SLR). In each scenario s, we assign retailers to facilities as follows. We first loop through all retailers i with $\sum_{j \in I} \hat{Y}_{ijs} \geq 1$ and assign i to the facility j with $\hat{Y}_{ijs} = 1$ that increases the objective function least based on the assignments made so far. Next we loop through retailers with $\sum_{j \in I} \hat{Y}_{ijs} = 0$ and assign each retailer to the open DC that increases the objective function least. In both cases we loop through retailers in decreasing order of mean demand μ_{is}. The resulting solution is feasible for (SLMRP) and its cost provides an upper bound on (3.1).

If the cost of the solution obtained using this procedure is less than the best upper bound found so far, we apply a retailer re-assignment heuristic to it. This heuristic is similar to that described by Daskin, Coullard, and Shen (2002) and involves re-assigning retailers from their currently assigned facility to a different one in a given scenario if doing so reduces the total cost. This is done for each scenario, since retailers may be assigned to different facilities in different scenarios. If at some point all of the demand assigned to a facility has been removed from the facility, one saves the fixed cost associated with the facility in addition to the other costs.

Daskin, Coullard, and Shen also describe a facility-exchange heuristic that involves swapping a facility out of the solution in favor of a facility not currently in the solution if doing so reduces the total cost; this procedure is similar to Teitz and Bart's (1968) procedure for the P-median problem. We did not use this heuristic in our computational tests because it is computationally expensive and the algorithm performed well without it.

3.2.3 Branch and Bound

If, when the Lagrangian procedure terminates, the best known lower bound is equal to the best known upper bound (to within some pre-specified tolerance), we have found the optimal solution to (SLMRP). Otherwise, a branch-and-bound procedure is employed to close the gap, with branching performed on the X_j (location) variables. At each node of the branch-and-bound tree, the facility selected for branching is the unfixed facility with the greatest assigned expected demand; if all facilities in the solution have already been forced open, we branch on an arbitrarily selected unforced facility. The variable is first forced to 0 and then to 1. Branching is done in a depth-first manner. The tree is fathomed at a given node if the lower bound at that node is greater than or equal to the objective value of the best feasible solution found anywhere in the tree to date, or if all facilities have been forced open or closed. In theory, if the overall lower bound is still not equal to the best upper bound found when the branch-and-bound procedure terminates, we must branch on the Y_{ijs} (assignment) variables, but this has never occurred in computational testing.

3.2.4 Variable Fixing

Suppose that the Lagrangian procedure terminates at the root node of the branch-and-bound tree with the lower bound strictly less than the upper bound. Let UB be the best upper bound found, let $\tilde{V}_j$ be the facility benefits under a particular set of Lagrange multipliers λ, and let LB be the lower bound (the objective value of (SLR)) under the same λ. Suppose further that $X_j = 0$ in the solution to (SLR) found using λ. If

$$\mathrm{LB} + \tilde{V}_j + f_j > \mathrm{UB} \tag{3.14}$$

then candidate site j *cannot* be part of the optimal solution, so we can fix $X_j = 0$. To see why this is true, imagine that we chose to branch on X_j. Clearly $\mathrm{LB} + \tilde{V}_j + f_j$ is a valid lower bound for the "$X_j = 1$" node (it would be the first lower bound found if we use λ as the initial multipliers at the new child node), so we would fathom the tree at this new node and never again consider setting $X_j = 1$.

Similarly, suppose $X_j = 1$ in the solution to (SLR) found using λ. If

$$\mathrm{LB} - (\tilde{V}_j + f_j) > \mathrm{UB} \tag{3.15}$$

then candidate site j *must* be part of the optimal solution, so we can fix $X_j = 1$. Note that in this case, $\tilde{V}_j + f_j < 0$ (otherwise we would have opened j), which is why the left-hand side might exceed UB.

We perform these variable-fixing checks twice after processing has terminated at the root node, once using the optimal multipliers λ and once using the most recent multipliers, as suggested by Daskin, Coullard, and Shen (2002). This procedure is quite effective in forcing variables open or closed because (SLR) tends to produce very tight lower bounds, making (3.14) or (3.15) hold for many facilities j. The time required to perform these checks is negligible.

3.2.5 Relationship to Weaver and Church's Algorithm

In Section 2.2.1.2, we discussed Weaver and Church's (1983) algorithm for the stochastic PMP. Essentially, their algorithm treats the multi-scenario PMP as a deterministic PMP with $|I||S|$ customers instead of $|I|$ and solves the resulting problem using Lagrangian relaxation. Our approach is very similar. The difference is that Weaver and Church solve the Lagrangian subproblem for each j, not segregating by scenario. In the SLMRP, this would amount to solving

$$\begin{array}{lll} \text{minimize} \quad \tilde{V}_j & = & \sum_{s\in S}\left[\sum_{i\in I} b_{is}Z_{is} + \sqrt{\sum_{i\in I} c_{is}Z_{is}}\right] \\ \text{subject to} & & Z_{is} \in \{0,1\} \qquad \forall i \in I, \forall s \in S \end{array}$$

for all j in place of problem (SSP$_{js}$), where $b_{is} = q_s\hat{d}_{ijs} - \lambda_{is}$ and $c_{is} = q_s^2\hat{K}_{js}^2\mu_{is}$, and $Z_{is} = 1$ if retailer i is assigned to facility j in scenario s. The difficulty with this problem is that the objective function contains $|S|$ square root terms instead of one, and our solution method relies on there being a single square root term. Fortunately, the problem given above decomposes by s, so we can solve the problem for each facility–scenario pair individually and sum over scenarios to obtain the benefit of each facility, as described above.

Similar reasoning explains why the DC–customer assignments must be scenario dependent in our model. If they were scenario independent, we would have assignment variables Y_{ij}, and the pricing problem would not decompose by s:

$$\begin{aligned} \text{minimize} \quad \tilde{V}_j &= \sum_{s\in S}\left[\sum_{i\in I} b_{is}Z_i + \sqrt{\sum_{i\in I} c_{is}Z_i}\right] \\ \text{subject to} \quad & Z_i \in \{0,1\} \qquad \forall i \in I \end{aligned}$$

The coefficients b_{is} and c_{is} are as defined above, but now $Z_i = 1$ if retailer i is assigned to facility j in *all scenarios*. Again, this problem has $|S|$ square root terms, making it much more difficult to solve. (An algorithm by Shu, Teo, and Shen (2001) can be extended to solve this problem, but it would require $O(n^{|S|}\log n)$ operations for each j, s.)

3.3 Multi-Commodity and Multi-Period Problems

Suppose we wanted to solve the LMRP for multiple commodities simultaneously. Since there are no capacity constraints, one might be tempted to aggregate the products and model them as one. But this strategy falsely assumes that risk-pooling benefits apply across products; that is, that holding inventory of one product protects against stockouts of another. However, the stochastic LMRP framework can be used to model this multi-commodity problem by letting S represent the set of products (instead of scenarios), letting $q_s = 1$ for all $s \in S$, and replacing f_j by $f_j/|S|$ in the objective function (3.1). The objective function (3.1) is then interpreted as adding the (product-independent) fixed location costs and the (product-dependent) transportation and inventory costs.

Constraints (3.2) say that each retailer must receive each product from exactly one DC (though it may receive different products from different DCs), and constraints (3.3) say that no retailer may receive any product from a DC that has not been opened. The solution method for the SLMRP is the same under this multi-commodity interpretation, except that now equation (3.13) is replaced by

$$\tilde{V}_j + f_j|S| < 0 \tag{3.16}$$

since the q_s now sum to $|S|$ instead of 1.

The problem with this interpretation is that it assumes each product is ordered individually, following a lot-sizing schedule based on the solutions to $|S|$ individual EOQ problems. Moreover, DCs pay fixed ordering and shipment costs (F and g) for each product, when in reality, DCs are likely to pay these costs for each order, regardless of the number of products ordered. In other words, this formulation does not take into account the economies of scale that can result from solving a multi-product lot-sizing problem to coordinate the ordering of multiple products simultaneously. Fortunately, our solution methodology requires only that the replenishment cost is concave in the aggregate demand served. This assumption appears to hold for a variety of multi-product lot-sizing problems; see, for example, the case problem mentioned in Chapter 8 of Chopra and Meindl (2001). Future research should focus on incorporating lot-sizing into the SLMRP framework.

This framework does allow us to model *tooling costs* t_{js} for stocking a given product s at a given DC j. Since the benefit $\tilde{V}_{js}$ is computed for each facility–product pair (j, s), we can add the tooling cost to $\tilde{V}_{js}$ and only use DC j for product s if the benefit is still negative. Then the definition of the benefit of facility j is

$$\tilde{V}_j = \sum_{s \in S} \min\{0, \tilde{V}_{js} + t_{js}\}. \tag{3.17}$$

Tooling costs are often encountered in practice and are frequently difficult to model since many supply chain design models do not already have binary variables for DC–product pairs. (See, for example, Geoffrion and Graves 1974, or Section 12.4 of Bramel and Simchi-Levi 1997.)

The SLMRP framework can also be used to model multi-period problems in which the parameters vary from period to period in a deterministic way. In this case, S is the set of time periods and parameter values are specified for each period. Again we set $q_s = 1$ for all s and replace f_j by $f_j/|S|$ for all j. Note that in this multi-period model, facilities are located before period 1, while assignments and inventory policies may change over time. It is not a truly dynamic model in which facilities may be opened or relocated over time. The tooling cost t_{js} could still be used in this case, representing a fixed cost for using a DC in a given time period, but it is unlikely that a firm would want to construct a DC but let it remain idle in any period. This similarly makes the tooling cost unnecessary in the standard SLMRP.

3.4 Computational Results

3.4.1 Experimental Design

We tested our algorithm for the SLMRP on the 49-node, 88-node, and 150-node data sets described by Daskin (1995). The 49-node data set represents the capitals of the lower

48 United States plus Washington, DC; the 88-node data set contains the 49-node data set plus the 50 largest cities in the 1990 U.S. census, minus duplicates; and the 150-node data set contains the 150 largest cities in the 1990 U.S. census.

For each data set, we generated 3-, 5-, and 9-scenario problems. We computed the "base" demand by dividing the population given by Daskin by 1000; these base demands were used to compute scenario-specific demands for 9 scenarios following the method described by Daskin, Hesse, and ReVelle (1997); in brief, this method involves defining an "attractor" point for each scenario and scaling each retailer's demand based on its distance to the attractor point. The total mean demand is the same in all scenarios for a given problem. The demand variance was set equal to the demand mean in all cases (i.e., $\gamma_s = 1$ for all s). Following Daskin, Hesse, and ReVelle, fixed location costs (f_j) were obtained by dividing the fixed cost given by Daskin by 10 for the 49-scenario problem and by 100 for the 88-node problem; for the 150-node problem, fixed costs for all retailers were set to 100. Fixed costs were chosen in this manner to provide a reasonable tradeoff between fixed costs and transportation and inventory costs. Retailer locations for scenario 1 were taken directly from Daskin for all three problems; for scenarios 2–9, the latitude and longitude values from scenario 1 were multiplied by a random number drawn uniformly from $U[0.95, 1.05]$. This has the effect of making the distances scenario specific. In all cases, great-circle distances were used.

As mentioned in Section 3.1, the ordering and shipping costs may be scenario-specific; we utilized this feature in our test problems. The fixed ordering and shipping costs (F_{js} and g_{js}, respectively) were set to 10 and the variable shipping cost (a_{js}) was set to 5 for all retailers in scenario 1. In scenarios 2–9, F_{js} and g_{js} were set to a random number drawn uniformly from $U[7.5, 12.5]$ and a_{js} was set to a random number drawn uniformly from $U[3.75, 6.25]$ (i.e., the costs were perturbed by up to 25% in either direction).

The holding cost h, the lead time L_j, and the days per year χ were set to 1. ($\chi = 1$ may seem unrealistic, but the weights β and θ can serve to translate daily parameters into yearly ones instead of χ.) z_α was set to 1.96 (guaranteeing at least a 97.5% service level). We tested five values of the weights β and θ.

The scenario probabilities for the 9-scenario problems are given by Daskin et al. (1997); they are: 0.01, 0.04, 0.15, 0.02, 0.34, 0.14, 0.09, 0.16, 0.05. To obtain the 3-scenario problem, we used the first 3 scenarios and scaled the probabilities so they sum to 1 (the new probabilities are 0.05, 0.2, 0.75), and similarly for the 5-scenario problem (obtaining probabilities 0.018, 0.071, 0.268, 0.036, 0.607).

The parameters used for the Lagrangian relaxation procedure are given in Table 3.1. For a more detailed description of these parameters, see Daskin (1995). The notation $\bar{\mu}$ in the table stands for the average mean demand, taken across all retailers and all scenarios. We terminated the branch-and-bound procedure when the optimality gap was less than 0.1%, or when 2,000 CPU seconds had elapsed.

We coded the algorithm in C++ and performed the computational tests on a Dell Inspiron 7500 notebook computer with a 500 MHz Pentium III processor and 128 MB memory.

Table 3.1: Parameters for Lagrangian relaxation procedure: SLMRP.

Parameter	Value
Maximum number of iterations at root node	1200
Maximum number of iterations at other nodes	400
Number of non-improving iterations before halving α	12
Initial value of α	2
Minimum value of α	0.00000001
Minimum LB–UB gap	0.1%
Initial value for λ_{is}	$10\bar{\mu} + 10f_i$

3.4.2 Algorithm Performance

Table 3.2 describes the algorithm's performance for our computational experiments. The columns are as follows.

 # Ret The number of retailers in the problem.

 # Scen The number of scenarios in the problem.

β The value of β.

θ The value of θ.

Overall LB The lower bound obtained from the branch-and-bound process.

Overall UB The objective value of the best feasible solution found during the branch-and-bound process.

Overall Gap The percentage difference between the overall upper and lower bounds.

Root LB The best lower bound obtained during the Lagrangian process at the root node.

Root UB The objective value of the best feasible solution found during the Lagrangian process at the root node.

Root Gap The percentage difference between the root-node upper and lower bounds.

 # Lag Iter The total number of Lagrangian relaxation iterations performed during the algorithm.

 # BB Nodes The number of branch-and-bound nodes explored during the algorithm.

CPU Time (sec.) The number of CPU seconds that elapsed before the algorithm terminated.

The optimal[1] solution was found (and proven to be optimal) at the root node in 29 out of 45 test problems. For the remaining problems, fewer than 10 branch-and-bound nodes were generally needed, though for a few problems more were necessary. In all but three cases, the optimality gap at the root node was less than 1%, and the root-node gap was always less than 3.1%, indicating that the bound provided by the Lagrangian relaxation process is very tight and that even without branch-and-bound, the Lagrangian procedure can be relied upon to generate a good feasible solution. For the two smaller data sets, the algorithm reached a provably optimal solution within the 2000-second limit in all but one case (in fact, in under two minutes in most cases). The algorithm's performance for the 150-node data set was slightly less impressive, with CPU times occasionally exceeding 2000 seconds and the algorithm terminating without a provably optimal solution. This is not surprising since these problems are quite large—for example, the 9-scenario problem

[1] If the optimality gap is less than or equal to 0.1%, we refer to the solution as optimal.

Table 3.2: SLMRP algorithm performance.

# Ret	# Scen	β	θ	Overall LB	Overall UB	Overall Gap	Root LB	Root UB	Root Gap	# Lag Iter	# BB Nodes	CPU Time (sec.)
49	3	0.001	0.1	149,741	149,888	0.10%	149,741	149,888	0.10%	123	1	5.7
49	5	0.001	0.1	151,161	151,302	0.09%	150,925	152,282	0.90%	3186	11	153.2
49	9	0.001	0.1	155,347	155,502	0.10%	154,945	159,655	3.04%	5754	19	486.2
49	3	0.005	0.1	303,327	303,626	0.10%	303,327	303,626	0.10%	44	1	2.9
49	5	0.005	0.1	303,527	303,827	0.10%	303,527	303,827	0.10%	205	1	12.5
49	9	0.005	0.1	316,541	316,847	0.10%	316,541	316,847	0.10%	70	1	10.1
49	3	0.005	0.5	312,658	312,970	0.10%	312,658	312,970	0.10%	76	1	5.4
49	5	0.005	0.5	312,191	312,500	0.10%	312,191	312,500	0.10%	188	1	11.9
49	9	0.005	0.5	325,883	326,205	0.10%	325,883	326,205	0.10%	120	1	15.8
49	3	0.005	1	321,280	321,421	0.04%	321,280	321,421	0.04%	37	1	2.6
49	5	0.005	1	320,041	320,351	0.10%	320,041	320,351	0.10%	87	1	7.9
49	9	0.005	1	334,340	334,666	0.10%	334,340	334,666	0.10%	63	1	8.1
49	3	0.005	20	498,378	498,775	0.08%	498,378	498,775	0.08%	76	1	6.0
49	5	0.005	20	493,292	493,771	0.10%	493,292	493,771	0.10%	104	1	7.4
49	9	0.005	20	512,453	512,953	0.10%	512,453	512,953	0.10%	198	1	20.2
88	3	0.001	0.1	25,318	25,320	0.01%	25,318	25,320	0.01%	979	1	58.6
88	5	0.001	0.1	24,487	24,509	0.09%	24,474	24,509	0.14%	1282	3	109.0
88	9	0.001	0.1	25,644	25,668	0.09%	25,597	26,065	1.83%	3578	13	529.6
88	3	0.005	0.1	55,928	55,983	0.10%	55,928	55,983	0.10%	523	1	36.9
88	5	0.005	0.1	54,067	54,120	0.10%	54,067	54,120	0.10%	729	1	71.9
88	9	0.005	0.1	57,998	58,025	0.05%	57,998	58,025	0.05%	1185	1	194.2
88	3	0.005	0.5	60,671	60,684	0.02%	60,671	60,684	0.02%	195	1	19.0
88	5	0.005	0.5	58,628	58,687	0.10%	58,628	58,687	0.10%	628	1	64.7
88	9	0.005	0.5	62,678	62,740	0.10%	62,497	62,740	0.39%	7299	27	1115.3
88	3	0.005	1	64,796	64,859	0.10%	64,796	64,859	0.10%	129	1	12.5
88	5	0.005	1	62,740	62,802	0.10%	62,740	62,802	0.10%	235	1	30.3
88	9	0.005	1	66,839	66,905	0.10%	66,803	66,986	0.27%	13506	53	> 2000.0
88	3	0.005	20	138,876	138,943	0.05%	138,876	138,943	0.05%	455	1	32.0
88	5	0.005	20	136,717	136,775	0.04%	136,717	136,862	0.11%	1041	3	103.1
88	9	0.005	20	142,272	142,370	0.07%	142,272	142,370	0.07%	328	1	66.8
150	3	0.001	0.1	14,847	14,917	0.47%	14,847	14,917	0.47%	15972	47	> 2000.0
150	5	0.001	0.1	15,141	15,155	0.09%	15,140	15,160	0.13%	8071	30	1319.7
150	9	0.001	0.1	15,794	16,210	2.63%	15,794	16,216	2.67%	5269	14	> 2000.0
150	3	0.005	0.1	23,739	23,763	0.10%	23,739	23,763	0.10%	1105	1	132.4
150	5	0.005	0.1	23,858	23,882	0.10%	23,817	23,882	0.27%	1890	5	340.4
150	9	0.005	0.1	24,137	24,161	0.10%	24,137	24,161	0.10%	1054	1	304.8
150	3	0.005	0.5	32,970	32,992	0.07%	32,969	33,016	0.14%	1223	3	157.7
150	5	0.005	0.5	33,044	33,077	0.10%	32,988	33,080	0.28%	1839	5	319.2
150	9	0.005	0.5	33,691	33,725	0.10%	33,691	33,725	0.10%	994	1	306.5
150	3	0.005	1	40,804	40,843	0.09%	40,804	40,843	0.09%	876	1	124.5
150	5	0.005	1	40,943	40,984	0.10%	40,876	40,987	0.27%	1831	5	309.9
150	9	0.005	1	41,876	41,918	0.10%	41,876	41,918	0.10%	997	1	322.5
150	3	0.005	20	155,653	155,781	0.08%	155,653	155,781	0.08%	227	1	37.0
150	5	0.005	20	157,380	157,508	0.08%	157,134	157,595	0.29%	3029	9	644.9
150	9	0.005	20	161,542	161,812	0.17%	161,369	161,983	0.38%	6107	18	> 2000.0

Table 3.3: SLMRP algorithm performance summary.

# Ret	# Scen	Avg. Overall Gap	Avg. CPU Time
49	3	0.08%	4.5
	5	0.10%	38.6
	9	0.10%	108.1
	Avg	0.09%	50.4
88	3	0.05%	31.8
	5	0.09%	75.8
	9	0.08%	797.9
	Avg	0.07%	301.8
150	3	0.16%	499.6
	5	0.09%	586.8
	9	0.62%	1011.5
	Avg	0.29%	699.3

has the equivalent of $|I||S| = 1350$ retailers. In addition, the size of the problem increases the time required at each iteration, hence the number of nodes that can be processed before the time limit is reached decreases as $|I|$ and $|S|$ increase.

The results in Table 3.2 are summarized in Table 3.3, in which the Overall Gap and CPU Time fields are averaged over β and θ and reported for each number of retailers and scenarios.

3.4.3 Variable Fixing and DC Locations

Table 3.4 gives information about the variable-fixing routine and the number of facilities opened. The first four columns are as described above. The other columns are as follows:

 # Fixed Open The number of facilities fixed open by the variable-fixing routine after processing at the root node.

 # Fixed Closed The number of facilities fixed closed by the variable-fixing routine after processing at the root node.

Total # Fixed The sum of the previous two columns.

Root Gap The percentage difference between the root-node upper and lower bounds, repeated here for reference.

 # Facil The number of facilities open in the best solution found.

As one might expect, the number of facilities forced into or out of the solution by the variable-fixing routine is larger for problems that have a smaller optimality gap at the root node. (Note that for problems that were solved to optimality at the root node, the variable-fixing routine is unnecessary; we performed it simply for completeness.)

For given values of β and θ, the number of DCs open in the optimal solution does not seem to increase or decrease with any regularity as the number of scenarios increases. However, it is evident that as β increases (from 0.001 to 0.005), the number of DCs increases. This is because when β is large, the transportation term becomes more significant in the objective function, making it desirable to have more DCs. Similarly, as θ increases, the number of DCs decreases because inventory becomes more expensive and risk-pooling becomes more attractive. These trends confirm results reported by Shen, Coullard, and Daskin (2003).

The results in Table 3.4 are summarized in Table 3.5, which reports, for each number of retailers and scenarios, the average percentage of facilities fixed open or closed by the variable-fixing routine and the average number of facilities open in the best solution found.

Table 3.4: SLMRP: variable-fixing and DC locations.

# Ret	# Scen	β	θ	# Fixed Open	# Fixed Closed	Total # Fixed	Root Gap	# Facil
49	3	0.001	0.1	6	36	42	0.10%	10
49	5	0.001	0.1	0	16	16	0.90%	11
49	9	0.001	0.1	0	4	4	3.04%	10
49	3	0.005	0.1	15	17	32	0.10%	30
49	5	0.005	0.1	8	14	22	0.10%	26
49	9	0.005	0.1	3	8	11	0.10%	30
49	3	0.005	0.5	12	15	27	0.10%	30
49	5	0.005	0.5	4	15	19	0.10%	26
49	9	0.005	0.5	3	9	12	0.10%	30
49	3	0.005	1	17	17	34	0.04%	30
49	5	0.005	1	7	15	22	0.10%	26
49	9	0.005	1	7	8	15	0.10%	30
49	3	0.005	20	9	18	27	0.08%	21
49	5	0.005	20	9	19	28	0.10%	22
49	9	0.005	20	5	10	15	0.10%	24
88	3	0.001	0.1	9	69	78	0.01%	13
88	5	0.001	0.1	1	59	60	0.14%	15
88	9	0.001	0.1	0	14	14	1.83%	16
88	3	0.005	0.1	14	31	45	0.10%	44
88	5	0.005	0.1	10	26	36	0.10%	41
88	9	0.005	0.1	15	30	45	0.05%	45
88	3	0.005	0.5	36	41	77	0.02%	41
88	5	0.005	0.5	6	29	35	0.10%	41
88	9	0.005	0.5	0	7	7	0.39%	43
88	3	0.005	1	14	29	43	0.10%	42
88	5	0.005	1	4	28	32	0.10%	40
88	9	0.005	1	0	9	9	0.27%	41
88	3	0.005	20	7	61	68	0.05%	19
88	5	0.005	20	2	55	57	0.11%	17
88	9	0.005	20	0	40	40	0.07%	20
150	3	0.001	0.1	0	0	0	0.47%	53
150	5	0.001	0.1	0	25	25	0.13%	52
150	9	0.001	0.1	0	0	0	2.67%	58
150	3	0.005	0.1	50	0	50	0.10%	133
150	5	0.005	0.1	4	0	4	0.27%	138
150	9	0.005	0.1	23	0	23	0.10%	146
150	3	0.005	0.5	16	0	16	0.14%	123
150	5	0.005	0.5	4	0	4	0.28%	124
150	9	0.005	0.5	11	0	11	0.10%	131
150	3	0.005	1	14	4	18	0.09%	108
150	5	0.005	1	3	0	3	0.27%	115
150	9	0.005	1	7	0	7	0.10%	127
150	3	0.005	20	1	0	1	0.08%	34
150	5	0.005	20	2	0	2	0.29%	33
150	9	0.005	20	0	0	0	0.38%	40

Table 3.5: SLMRP: variable-fixing and DC locations summary.

# Ret	# Scen	Avg. % Fixed	Avg. # Facil
49	3	66.1%	24.2
	5	43.7%	22.2
	9	23.3%	24.8
	Avg	44.4%	23.7
88	3	70.7%	31.8
	5	50.0%	30.8
	9	26.1%	33.0
	Avg	48.9%	31.9
150	3	11.3%	90.2
	5	5.1%	92.4
	9	5.5%	100.4
	Avg	7.3%	94.3

3.4.4 Stochastic vs. Deterministic Solutions

Table 3.6 indicates the differences between the stochastic (i.e., min-expected-cost) solutions and the individual scenario solutions. The first four columns are as described above, and the remaining columns are as follows:

 # DCs Different The average, minimum, and maximum (across scenarios) number of DC locations in the stochastic solution that are different from locations in the single-scenario solutions, computed as the number of facilities in the stochastic solution that are not in the scenario solution plus the number of facilities in the scenario solution that are not in the stochastic solution.

% Regret The average, minimum, and maximum (across scenarios) percentage regret that would result from implementing the best stochastic solution found instead of the optimal solution for a given scenario.

 # Scen-Spec Assign The number of retailers that are assigned to different DCs in different scenarios in the best stochastic solution found.

Clearly, the stochastic solution and the single-scenario solutions differ substantially in their choices of DC locations. This suggests that each of the single-scenario solutions would perform poorly in long-run expected cost. Furthermore, quite a few retailers—roughly half on average, but up to 97%—are assigned to different DCs in different scenarios, indicating the value of allowing retailer assignments to be scenario dependent. Finally, we note that implementing the stochastic solution will entail roughly 8% regret on average and nearly 25% regret in the worst case. This suggests the need for a way to bound the regret while still minimizing expected cost; the model presented in the next chapter does just that.

The results in Table 3.6 are summarized in Table 3.7, which reports, for each number of retailers and scenarios, the averages across β and θ of the "Avg # DCs Different," "Avg % Regret," and "# Scen-Spec Assign" columns.

3.5 Chapter Summary

In this chapter we extended the LMRP to handle stochastic demands and costs. The random parameters are described by discrete scenarios, each with a known probability of occurrence; the objective is to minimize total expected cost. Facility location decisions must be made in the first stage, but allocation and inventory decisions are made in the second stage, after the uncertainties have been resolved. The model can also be used to solve multi-commodity and multi-period problems. We presented a Lagrangian-

Table 3.6: SLMRP: stochastic vs. deterministic solutions.

# Ret	# Scen	β	θ	# DCs Different Avg	# DCs Different Min	# DCs Different Max	% Regret Avg	% Regret Min	% Regret Max	# Scen-Spec Assign
49	3	0.001	0.1	8.3	1	13	6.5%	0.5%	12.1%	29
49	5	0.001	0.1	12.4	8	16	10.4%	3.9%	21.2%	36
49	9	0.001	0.1	9.0	5	13	7.2%	4.1%	11.1%	36
49	3	0.005	0.1	7.3	4	9	5.7%	0.6%	8.2%	18
49	5	0.005	0.1	11.2	2	16	10.2%	0.9%	16.6%	23
49	9	0.005	0.1	10.6	6	14	9.5%	4.9%	18.9%	19
49	3	0.005	0.5	9.0	4	13	5.5%	0.7%	8.1%	18
49	5	0.005	0.5	11.2	3	16	9.8%	0.9%	16.1%	23
49	9	0.005	0.5	10.7	7	13	9.3%	4.9%	18.4%	19
49	3	0.005	1	10.0	7	13	5.4%	0.8%	8.1%	18
49	5	0.005	1	11.2	3	15	9.5%	0.9%	15.6%	23
49	9	0.005	1	11.7	7	16	9.2%	4.9%	18.0%	19
49	3	0.005	20	10.0	1	16	8.4%	0.1%	14.3%	28
49	5	0.005	20	12.2	8	16	7.1%	2.3%	12.3%	29
49	9	0.005	20	12.4	8	20	7.1%	3.6%	11.8%	32
88	3	0.001	0.1	10.0	2	15	7.7%	0.1%	12.4%	48
88	5	0.001	0.1	14.0	8	20	8.5%	2.2%	12.6%	61
88	9	0.001	0.1	13.1	8	16	7.6%	2.6%	12.3%	70
88	3	0.005	0.1	16.0	4	26	8.5%	0.6%	15.4%	36
88	5	0.005	0.1	22.2	11	25	9.9%	1.8%	15.5%	47
88	9	0.005	0.1	22.8	16	29	11.1%	5.6%	17.9%	43
88	3	0.005	0.5	16.7	5	28	8.5%	0.5%	17.2%	38
88	5	0.005	0.5	22.2	12	28	9.0%	1.9%	14.4%	47
88	9	0.005	0.5	23.2	18	30	10.5%	6.2%	17.5%	46
88	3	0.005	1	18.0	10	24	8.3%	0.5%	16.6%	37
88	5	0.005	1	22.2	11	26	8.8%	2.0%	15.5%	49
88	9	0.005	1	23.0	16	33	10.5%	5.2%	19.9%	51
88	3	0.005	20	12.7	5	18	3.4%	0.4%	5.9%	56
88	5	0.005	20	14.6	7	20	5.4%	0.7%	8.1%	65
88	9	0.005	20	16.7	12	23	5.8%	2.7%	9.8%	75
150	3	0.001	0.1	37.0	14	54	10.6%	1.3%	20.7%	94
150	5	0.001	0.1	42.4	21	53	10.4%	1.0%	20.8%	105
150	9	0.001	0.1	43.7	33	51	13.2%	5.7%	23.7%	109
150	3	0.005	0.1	33.7	10	64	10.2%	0.8%	24.4%	31
150	5	0.005	0.1	28.0	17	55	8.2%	2.1%	22.2%	26
150	9	0.005	0.1	24.9	16	55	5.6%	2.8%	19.9%	23
150	3	0.005	0.5	38.3	14	63	9.2%	0.8%	20.2%	60
150	5	0.005	0.5	35.0	24	56	7.5%	1.8%	19.0%	66
150	9	0.005	0.5	32.9	23	57	6.7%	2.8%	18.8%	73
150	3	0.005	1	43.3	21	65	8.8%	0.6%	19.6%	73
150	5	0.005	1	38.6	21	58	6.7%	1.6%	14.6%	75
150	9	0.005	1	38.6	29	58	6.5%	3.0%	16.1%	83
150	3	0.005	20	25.7	13	40	1.6%	0.8%	3.0%	130
150	5	0.005	20	29.4	16	39	2.8%	0.7%	5.1%	134
150	9	0.005	20	34.9	25	48	2.8%	1.2%	5.4%	146

Table 3.7: SLMRP: stochastic vs. deterministic solutions summary.

# Ret	# Scen	Avg # DCs Different	Avg % Regret	Avg # Scen-Spec Assign
49	3	8.9	6.3%	22.2
	5	11.6	9.4%	26.8
	9	10.9	8.5%	25.0
	Avg	10.5	8.0%	24.7
88	3	14.7	7.3%	43.0
	5	19.0	8.3%	53.8
	9	19.8	9.1%	57.0
	Avg	17.8	8.2%	51.3
150	3	35.6	8.1%	77.6
	5	34.7	7.1%	81.2
	9	35.0	7.0%	86.8
	Avg	35.1	7.4%	81.9

relaxation–based algorithm for solving the stochastic LMRP. Optimal solutions to this problem will have the best possible long-run expected cost but its performance may be quite variable across scenarios. Many decision makers want solutions that will perform well regardless of which scenario comes to pass—they want the regret across scenarios to be bounded while still keeping the expected cost down. The model we present in the next chapter addresses this problem.

Chapter 4

The p-Robust Stochastic Location Model with Risk Pooling (p-SLMRP)

The stochastic LMRP (SLMRP) discussed in the previous chapter seeks to minimize the total expected cost of a supply chain network across all scenarios. The optimal solution may be excellent for some scenarios but quite poor for others. In many situations, decision makers are evaluated *ex post*, after the uncertainty has been resolved and costs have been realized. In such situations, decision makers are often motivated to seek minimax regret solutions that appear effective no matter which scenario is realized. The robustness measure we discuss in this section combines the advantages of both the min-expected-cost and minimax regret measures by seeking the least-cost solution (in the expected value) that bounds the regret by a pre-specified limit.

Let us make this concept more rigorous. Let (P_s) be a deterministic (i.e., single-scenario) optimization problem, indexed by the scenario index s. (That is, for each scenario s, there is a different problem (P_s). The structure of these problems is identical; only the data are different.) For each $s \in S$, let z_s^* be the optimal objective value for (P_s).

Definition 4.1 *Let $p \geq 0$ be a constant. Let X be a solution to (P_s) for all $s \in S$, and let $z_s(X)$ be the objective value of problem (P_s) under solution X. X is called* p-robust *if for all $s \in S$,*

$$\frac{z_s(X) - z_s^*}{z_s^*} \leq p \tag{4.1}$$

or, equivalently,

$$z_s(X) \leq (1 + p) z_s^*. \tag{4.2}$$

We will say that scenario s is *p-feasible* for a given solution X if equation (4.2) holds and *p-infeasible* otherwise.

The notion of p-robustness was introduced by Kouvelis, Kurawarwala, and Gutiérrez (1992) and used subsequently in several other papers (see Section 2.2.4 for a description). These papers do not refer to this robustness measure as p-robustness, but simply as "robustness." We will adopt the term "p-robustness" to distinguish this robustness measure from others.

In this chapter, we formulate and solve the problem of finding the minimum-expected-cost p-robust solution to the stochastic LMRP. We call this problem the p-SLMRP. Since the LMRP reduces to the UFLP when $\theta = 0$, our method also solves a p-robust version of the stochastic UFLP. As discussed below in Section 4.3, this method can be used iteratively as a heuristic to solve the minimax regret LMRP or UFLP.

Our Lagrangian relaxation algorithm for the p-SLMRP cannot provide lower bounds that are tighter than the continuous relaxation bound because the Lagrangian subproblem has the integrality property. In Section 4.4, we show that two classical facility location problems, the P-median problem and the uncapacitated fixed-charge location problem, can be solved using a variable-splitting method whose subproblem does *not* have the integrality property, resulting in tighter theoretical bounds.

4.1 Formulation

To formulate the p-SLMRP, we need to introduce two additional parameters:

Parameters

p = the desired bound on the relative regret of a solution, $p \geq 0$

z_s^* = the optimal objective value of problem (LMRP) under the data from scenario s, for $s \in S$

The optimal scenario objectives z_s^* are inputs to the model; they may be read in as data or computed in a pre-processing step using any algorithm for the LMRP. We need to add one additional set of constraints to (SLMRP) to ensure p-robustness. These constraints appear in the formulation below as (4.6). (Throughout this chapter, we continue to assume that Assumption 3.1 holds.)

$$(p\text{-SLMRP})\quad \text{minimize} \quad \sum_{s\in S}\sum_{j\in I} q_s \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + \hat{K}_{js}\sqrt{\sum_{i\in I}\mu_{is}Y_{ijs}} \right\} \tag{4.3}$$

$$\text{subject to} \quad \sum_{j\in I} Y_{ijs} = 1 \qquad \forall i\in I, \forall s\in S \tag{4.4}$$

$$Y_{ijs} \le X_j \qquad \forall i\in I, \forall j\in I, \forall s\in S \tag{4.5}$$

$$\sum_{j\in I} \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + \hat{K}_{js}\sqrt{\sum_{i\in I}\mu_{is}Y_{ijs}} \right\} \le (1+p)z_s^* \qquad \forall s\in S \tag{4.6}$$

$$X_j \in \{0,1\} \qquad \forall j\in I \tag{4.7}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i\in I, \forall j\in I, \forall s\in S \tag{4.8}$$

Constraints (4.6) require the relative regret of the solution in each scenario to be no greater than p. The objective function and the other constraints are as described in Section 3.1. Since the p-SLMRP reduces to the SLMRP when p is large, it is NP-hard.

For convenience, define

$$z_s(X,Y) = \sum_{j\in I} \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} + \hat{K}_{js}\sqrt{\sum_{i\in I}\mu_{is}Y_{ijs}} \right\}.$$

$z_s(X,Y)$ is the cost of solution (X,Y) in scenario s. The objective function (4.3) can be re-written as

$$\sum_{s\in S} q_s z_s(X,Y) \tag{4.9}$$

and constraints (4.6) can be re-written as

$$z_s(X,Y) \le (1+p)z_s^* \qquad \forall s\in S. \tag{4.10}$$

4.2 Solution Procedure

As in the solution procedure for the SLMRP, we will use Lagrangian relaxation to find a lower bound for (p-SLMRP), relaxing both the assignment constraints and the p-robustness constraints. Several complications arise when solving this problem that do not arise in the SLMRP. First, finding an upper bound is not trivial, since for small p, it may be difficult to find a feasible solution, and for even smaller p, none may exist. Second, the subgradient optimization procedure relies on having an upper bound at each iteration for use in the numerator of the step-size calculation, so we need some proxy if no feasible solution has yet been found. Finally, it may not be clear from the outset whether the problem is infeasible, so we need some mechanism for detecting infeasibility, otherwise the branch-and-bound procedure may explore the entire tree before terminating. We will resolve these issues in the following sections.

4.2.1 Lower Bound

4.2.1.1 Lagrangian Relaxation

If we relax constraints (4.4) and (4.6), we get the following problem:

$$
\begin{aligned}
(p\text{-SLR}) \quad & \underset{\lambda,\pi \geq 0}{\text{maximize}} \ \mathcal{L}_{\lambda,\pi} = \\
& \underset{X,Y}{\text{minimize}} \quad \sum_{s\in S} z_s(X,Y) + \sum_{s\in S}\sum_{i\in I}\lambda_{is}\left(1-\sum_{j\in I}Y_{ijs}\right) \\
& \qquad + \sum_{s\in S}\pi_s\left[z_s(X,Y)-(1+p)z_s^*\right] \\
& \quad = \sum_{s\in S}\sum_{j\in I}\Bigg\{(q_s+\pi_s)f_jX_j + \sum_{i\in I}\Big[(q_s+\pi_s)\hat{d}_{ijs}-\lambda_{is}\Big]Y_{ijs} \\
& \qquad + (q_s+\pi_s)\hat{K}_{js}\sqrt{\sum_{i\in I}\mu_{is}Y_{ijs}}\Bigg\} + \sum_{s\in S}\sum_{i\in I}\lambda_{is} - \sum_{s\in S}\pi_s(1+p)z_s^* \\
& \quad = \sum_{s\in S}\sum_{j\in I}\Bigg\{\tilde{f}_{js}X_j + \sum_{i\in I}\tilde{d}_{ijs}Y_{ijs} + \tilde{K}_{js}\sqrt{\sum_{i\in I}\mu_{is}Y_{ijs}}\Bigg\} + C \qquad (4.11)
\end{aligned}
$$

$$
\begin{aligned}
\text{subject to} \quad & Y_{ijs} \leq X_j \qquad \forall i \in I, \forall j \in I, \forall s \in S \qquad (4.12) \\
& X_j \in \{0,1\} \qquad \forall j \in I \qquad (4.13) \\
& Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \qquad (4.14)
\end{aligned}
$$

The multipliers λ and π correspond to constraints (4.4) and (4.6), respectively. The notation in the last line of the objective function is as follows:

$$
\begin{aligned}
\tilde{f}_{js} &= (q_s+\pi_s)f_j \\
\tilde{d}_{ijs} &= (q_s+\pi_s)\hat{d}_{ijs} - \lambda_{is} \\
\tilde{K}_{js} &= (q_s+\pi_s)\hat{K}_{js} \\
C &= \sum_{s\in S}\sum_{i\in I}\lambda_{is} - \sum_{s\in S}\pi_s(1+p)z_s^*
\end{aligned}
$$

As before, we can restrict $\lambda \geq 0$. Since (4.6) is a "less-than-or-equal-to" constraint in a minimization problem, the multipliers π are also restricted to be non-negative. The constraints are identical to their counterparts in (SLR). This problem has the same form as (SLR) and can be solved in the same manner. (p-SLR) decomposes by j and s. For given $j \in I,\ s \in S$, we solve

$$(p\text{-SSP}_{js}) \qquad \text{minimize} \quad \tilde{V}_{js} = \sum_{i \in I} b_i Z_i + \sqrt{\sum_{i \in I} c_i Z_i} \tag{4.15}$$

$$\text{subject to} \quad Z_i \in \{0,1\} \qquad \forall i \in I \tag{4.16}$$

where

$$b_i = \tilde{d}_{ijs}$$

$$c_i = \tilde{K}^2_{js} \mu_{is}$$

$$Z_i = Y_{ijs}$$

For given j, s, we can solve this problem using the technique described in Section 2.5.2 and make open/close decisions as described in Section 3.2.1. In particular, we open a DC at location j if its benefit exceeds its fixed cost, that is, if

$$\tilde{V}_j + \tilde{f}_j < 0,$$

where $\tilde{V}_j = \sum_{s \in S} \tilde{V}_{js}$ and $\tilde{f}_j = \sum_{s \in S} \tilde{f}_{js} = \left(1 + \sum_{s \in S} \pi_s\right) f_j$.

4.2.1.2 Subgradient Optimization

Subgradient optimization has been widely and successfully used to update multipliers in Lagrangian relaxation algorithms for location problems, and the details of the method are usually similar to those outlined by Fisher (1981, 1985) or Daskin (1995). For the p-SLMRP, however, the performance of our algorithm can be improved by modifying the standard subgradient optimization procedure.

Assume for now that at the current iteration (say n) of the Lagrangian procedure, at least one feasible solution has been found and that UB is the objective value of the best solution found. (In Section 4.2.2 we describe what to do if no feasible solution has been found as of iteration n.) Let $\mathcal{L}^n$ be the value of the lower bound found during iteration n and let (X^n, Y^n) be the optimal solution to problem (p-SLR) at iteration n. The standard subgradient step-size t^n is computed as follows:

$$t^n = \frac{\alpha^n(\mathrm{UB} - \mathcal{L}^n)}{\sum_{s\in S}\sum_{i\in I}\left(1 - \sum_{j\in I} Y_{ijs}^n\right)^2 + \sum_{s\in S}\left(z_s(X,Y) - (1+p)z_s^*\right)^2} \tag{4.17}$$

where α^n is a constant on the nth iteration. Typically, α^1 is set to a given value and halved after a given number R of consecutive iterations have failed to improve the lower bound $\mathcal{L}^n$; we use $\alpha^1 = 2$ and $R = 20$ in our computational testing. The denominator sums the squared deviations of both sets of relaxed constraints. The Lagrange multipliers are updated as follows:

$$\lambda_{is}^{n+1} = \max\left\{0, \lambda_{is}^n + t^n\left(1 - \sum_{j\in I} Y_{ijs}^n\right)\right\}$$

$$\pi_s^{n+1} = \max\left\{0, \pi_s^n + t^n\left(c_s(X^n, Y^n) - (1+p)z_s^*\right)\right\}$$

Unfortunately, the denominator in (4.17) is badly scaled because the first term is on the order of magnitude of the number of retailers times the number of scenarios (typically in the hundreds), while the second is on the order of magnitude of the solution cost squared (typically in the millions or more). This leads to slower than desired convergence of the Lagrangian procedure. To avoid this problem, we "normalize" the second term by dividing constraints (4.6) by νz_s^* before solving (p-SLR), where $\nu > 0$ is a constant, yielding the following constraint:

$$\frac{1}{\nu z_s^*} z_s(X, Y) \leq \frac{1+p}{\nu}.$$

The notation in the objective function (4.11) then becomes:

$$\begin{aligned}
\tilde{f}_{js} &= \left(q_s + \frac{\pi_s}{\nu z_s^*}\right) f_j \\
\tilde{d}_{ijs} &= \left(q_s + \frac{\pi_s}{\nu z_s^*}\right) \hat{d}_{ijs} - \lambda_{is} \\
\tilde{K}_{js} &= \left(q_s + \frac{\pi_s}{\nu z_s^*}\right) \hat{K}_{js} \\
C &= \sum_{s \in S} \sum_{i \in I} \lambda_{is} - \sum_{s \in S} \frac{\pi_s}{\nu}(1+p)
\end{aligned}$$

The problem is solved in the same manner as before. The step-size calculation (4.17) now becomes:

$$t^n = \frac{\alpha^n(\mathrm{UB} - \mathcal{L}^n)}{\sum_{s \in S} \sum_{i \in I} \left(\sum_{j \in I} 1 - Y_{ijs}^n\right)^2 + \sum_{s \in S} \left(\frac{z_s(X,Y)}{\nu z_s^*} - \frac{1+p}{\nu}\right)^2} \tag{4.18}$$

We discuss our method for choosing a value for ν in Section 4.5.1.2.

A similar adjustment was used by Beasley (1993) and by Marín and Pelegrín (1999), both in the context of capacitated location problems, and by Nozick (2001) in the context of a multi-objective model that is equivalent to the UFLP with an aggregated capacity constraint. These authors scale the capacity constraints by dividing them by the right-hand side.

Another way to handle the difference in scale between the two sets of relaxed constraints is to allow each set to have its own step size during the subgradient optimization procedure. In particular, we can compute step sizes t_λ^n and t_π^n for λ and π, respectively, as

$$t_\lambda^n = \frac{\alpha^n(\mathrm{UB} - \mathcal{L}^n)}{\sum_{s\in S}\sum_{i\in I}\left(\sum_{j\in I} 1 - Y_{ijs}^n\right)^2} \tag{4.19a}$$

$$t_\pi^n = \frac{\alpha^n(\mathrm{UB} - \mathcal{L}^n)}{\sum_{s\in S}\left(z_s(X,Y) - (1+p)z_s^*\right)^2} \tag{4.19b}$$

Note that if the step sizes are separated as in (4.19), there is no need to divide (4.6) by νz_s^* since the difference in scales is only a problem when they appear together in the denominator of a pooled step size. We have not found any previous use of this idea in the literature when more than one set of constraints is relaxed in a Lagrangian relaxation algorithm. We present a comparison of the two step-size calculations, as well as of different values of ν, in Section 4.5.

4.2.2 Detecting Infeasibility

The procedure outlined in the previous section assumes that we have found a feasible solution with objective value UB. But finding a p-robust solution is not always easy, even if such solutions exist. Let

$$Q = \sum_{s\in S} q_s(1+p)z_s^*. \tag{4.20}$$

Theorem 4.1 *If (p-SLMRP) is feasible, then Q is an upper bound on its optimal objective value.*

Proof. Let (X^*, Y^*) be an optimal solution for (p-SLMRP). The objective value under solution (X^*, Y^*) is

$$\sum_{s\in S} q_s z_s(X^*, Y^*) \le \sum_{s\in S} q_s(1+p)z_s^* = Q$$

by constraints (4.6). □

Theorem 4.1 has two important uses. First, if no feasible solution has been found as of iteration n, we set UB $= Q$ in the step-size calculation (4.17), (4.18), or (4.19) in the subgradient optimization procedure. Second, we can use Theorem 4.1 to detect when problem (p-SLMRP) is infeasible. In particular, if the Lagrangian procedure and/or the branch-and-bound procedure yield a lower bound greater than Q, we can terminate the procedure and conclude that the problem is infeasible. One would like the Lagrangian procedure to yield bounds greater than Q *whenever* (p-SLMRP) is infeasible, providing a test for feasibility in every case. Unfortunately, there are infeasible instances for which the Lagrangian bound is less than Q. In the next section, we investigate the circumstances under which we can expect to find λ, π such that $\mathcal{L}_{\lambda,\pi} > Q$.

4.2.3 Unboundedness of (p-SLR)

Let ($\overline{p\text{-SLMRP}}$) be the continuous relaxation of (p-SLMRP), that is, (p-SLMRP) with constraints (4.7) and (4.8) replaced by

$$0 \le X_j \le 1 \qquad \forall j \in I \tag{4.21}$$

$$0 \le Y_{ijs} \le 1 \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.22}$$

It is possible that (p-SLMRP) is infeasible but ($\overline{p\text{-SLMRP}}$) is feasible. When this is the case, the optimal objective value of (p-SLR) is equal to that of ($\overline{p\text{-SLMRP}}$) (since it has the integrality property), which is less than or equal to Q. It is our conjecture that if ($\overline{p\text{-SLMRP}}$) is infeasible, then ($p$-SLR) is unbounded, meaning that we can always find

$\lambda,\ \pi$ such that $\mathcal{L}_{\lambda,\pi} > Q$.

Conjecture 4.1 *If ($\overline{p\text{-SLMRP}}$) is infeasible, then (p-SLR) is unbounded; that is, for any $M \in \mathbb{R}$, there exists $\lambda,\ \pi$ such that $\mathcal{L}_{\lambda,\pi} > M$.*

We can prove this conjecture for *linear* programs, including the ($\overline{p\text{-SLMRP}}$) when $\theta = 0$—the LP relaxation of the p-robust stochastic UFLP:

Theorem 4.2 *Let (P) be a linear program of the form*

$$
\begin{aligned}
\text{(P)} \quad \text{minimize} \quad & cx \\
\text{subject to} \quad & Ax = b \\
& Dx \le e \\
& x \ge 0
\end{aligned}
$$

with $c \ge 0$, and let (LR) be the Lagrangian relaxation obtained by relaxing the constraints $Ax = b$. If (P) is infeasible, then (LR) is unbounded.

Proof. Suppose (P) is infeasible. The linear programming dual of (P) is given by

$$
\begin{aligned}
\text{(D)} \quad \text{maximize} \quad & -bu + ev \\
\text{subject to} \quad & -uA + vD \le c \\
& u \text{ unrestricted} \\
& v \le 0
\end{aligned}
$$

(In standard form, the coefficients of u would be $+1$ instead of -1, but since u is unrestricted, we can replace u with $-u$ in the formulation.) From standard duality theory, we know that when (P) is infeasible, (D) is either infeasible or unbounded. Since $c \geq 0$, (D) is always feasible because $u = v = 0$ is a solution; therefore if (P) is infeasible, (D) is unbounded.

The Lagrangian relaxation of (P) is given by

$$\begin{array}{llll} \text{(LR)} & \underset{u}{\text{maximize}} & \underset{x}{\text{minimize}} & cx + u(Ax - b) \\ & & \text{subject to} & Dx \leq e \\ & & & x \geq 0 \end{array}$$

where u is the vector of Lagrange multipliers. We need to show that for any $M \in \mathbb{R}$, there exists some u^* such that

$$\min_x \{cx + u^*(Ax - b) | Dx \leq e, x \geq 0\} > M.$$

Since (D) is unbounded, there exists $u^*,\ v^*$ such that $-bu^* + ev^* > M$, $-u^*A + v^*D \leq c$, and $v^* \leq 0$. Then

$$\begin{aligned} & \min_x \{cx + u^*(Ax - b) | Dx \leq e, x \geq 0\} \\ = & \min_x \{(c + u^*A)x - u^*b | Dx \leq e, x \geq 0\} \\ = & \max_v \{ve - u^*b | vD \leq c + u^*A, v \leq 0\} && \text{(by strong duality)} \\ = & \max_v \{-u^*b + ve | -u^*A + vD \leq c, v \leq 0\} \\ \geq & -u^*b + v^*e && \text{(since } v^* \text{ is feasible for (D))} \\ > & M && \text{(by choice of } u^*, v^*\text{)} \end{aligned}$$

Therefore (LR) is unbounded. □

The upshot of Theorem 4.2 is that when the objective function has non-negative costs, the Lagrangian dual behaves like the LP dual in the sense that when the problem is infeasible, the dual problem is unbounded. We have not been able to locate this result in the literature, although surely it must have been proven previously; we do not claim that this is the first proof of Theorem 4.2 to appear.

Let $(\overline{p\text{-SLMRP}}^0)$ be the problem attained by setting $\theta = 0$ in $(\overline{p\text{-SLMRP}})$:

$$(\overline{p\text{-SLMRP}}^0) \quad \text{minimize} \quad \sum_{s\in S}\sum_{j\in I} q_s \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} \right\} \tag{4.23}$$

$$\text{subject to} \quad \sum_{j\in I} Y_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{4.24}$$

$$Y_{ijs} \le X_j \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.25}$$

$$\sum_{j\in I} \left\{ f_j X_j + \sum_{i\in I} \hat{d}_{ijs} Y_{ijs} \right\} \le (1+p) z_s^* \qquad \forall s \in S \tag{4.26}$$

$$0 \le X_j \le 1 \qquad \forall j \in I \tag{4.27}$$

$$0 \le Y_{ijs} \le 1 \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.28}$$

Note that $(\overline{p\text{-SLMRP}}^0)$ is the LP relaxation of a p-robust version of the UFLP.

Corrolary 4.1 *If* $(\overline{p\text{-}SLMRP}^0)$ *is infeasible, then the Lagrangian relaxation obtained by relaxing constraints* (4.24) *and* (4.26) *is unbounded.*

The proof of this corollary is immediate by applying Theorem 4.2. Although constraints (4.26) are inequality constraints, not equality constraints as in the statement of Theorem 4.2, they can be converted to equality constraints through the use of slack variables.

Theorem 4.2 applies only to linear programs, not to non-linear programs like $(\overline{p\text{-SLMRP}})$. If $(\overline{p\text{-SLMRP}}^0)$ were always infeasible when $(\overline{p\text{-SLMRP}})$ is infeasible, Conjecture 4.1

Table 4.1: Facility data for example problem.

Parameter	Value Facility 1	Facility 2
Fixed cost (f_j)	2	2
Distance to retailer, scenario 1	1	2
Distance to retailer, scenario 2	2	1

Table 4.2: Other data for example problem.

Parameter	Value
Demand mean, scenarios 1 and 2 (μ_s)	1
Demand variance, scenarios 1 and 2 (σ_s^2)	1
Variance-to-mean ratio, scenarios 1 and 2 (γ_s)	1
β	1
θ	1
z_α	1
Holding cost (h)	1
Lead time (L)	1

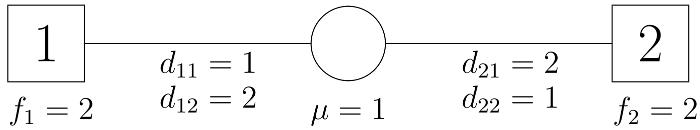
would hold because the objective function value of the Lagrangian relaxation of ($\overline{p\text{-SLMRP}}$) is greater than or equal to that of the Lagrangian relaxation of ($\overline{p\text{-SLMRP}}^0$). Unfortunately, however, it is possible that ($\overline{p\text{-SLMRP}}$) is infeasible but its "linear" version ($\overline{p\text{-SLMRP}}^0$) is feasible.

4.2.4 Example

In this section we present an example in which Conjecture 4.1 holds. Consider an instance of (p-SLMRP) with two potential facility locations and one retailer. There are two scenarios with equal probability ($q_1 = q_2 = \frac{1}{2}$). The data for the problem are given in Tables 4.1 and 4.2. All costs not listed in the table are equal to 0. Note that the distances are scenario dependent. The example instance is pictured in Figure 4.1.

Using the parameters in Table 4.2, we get $\hat{K}_{js} = 1$ for all j, s. In scenario 1, the optimal solution is to locate at facility 1, and in scenario 2 it is to locate at facility 2.

Figure 4.1: An infeasible (p-SLMRP) instance with unbounded (p-SLR).



Optimal scenario solution costs are $z_1^* = z_2^* = 4$. In both cases the regret for locating at the "wrong" facility is 1. Let $p = 0.01$; then the right-hand side of constraints (4.6) is 4.04. For simplicity, remove the index i from the allocation variables; $Y_{js} = 1$ if the single retailer is served by facility j in scenario s. Then (p-SLMRP) is as follows:

$$
\begin{aligned}
(p\text{-SLMRP}) \quad \text{minimize} \quad & 2X_1 + 2X_2 + \tfrac{1}{2}\Big[Y_{11} + 2Y_{12} + 2Y_{21} + Y_{22} \\
& + \sqrt{Y_{11}} + \sqrt{Y_{12}} + \sqrt{Y_{21}} + \sqrt{Y_{22}}\Big] && (4.29) \\
\text{subject to} \quad & Y_{11} + Y_{21} = 1 && (4.30) \\
& Y_{12} + Y_{22} = 1 && (4.31) \\
& Y_{11} \le X_1 && (4.32) \\
& Y_{21} \le X_2 && (4.33) \\
& Y_{12} \le X_1 && (4.34) \\
& Y_{22} \le X_2 && (4.35) \\
& 2X_1 + 2X_2 + Y_{11} + 2Y_{21} + \sqrt{Y_{11}} + \sqrt{Y_{21}} \le 4.04 && (4.36) \\
& 2X_1 + 2X_2 + 2Y_{12} + Y_{22} + \sqrt{Y_{12}} + \sqrt{Y_{22}} \le 4.04 && (4.37) \\
& X, Y \in \{0, 1\} && (4.38)
\end{aligned}
$$

This problem is infeasible: constraint (4.36) implies $X_1 = Y_{11} = 1$, but then constraint (4.37) is impossible to satisfy. The reader can easily verify that its continuous relaxation, too, is infeasible. Since $Y_{js} = 0$ or 1, we can replace $\sqrt{Y_{js}}$ with Y_{js} throughout. The

Lagrangian subproblem is as follows:

$$\begin{aligned}
(p\text{-SLR}) \quad & \underset{\lambda,\pi\geq 0}{\text{maximize}} \quad \underset{X,Y}{\text{minimize}} \quad 2X_1 + 2X_2 + Y_{11} + \tfrac{3}{2}Y_{12} + \tfrac{3}{2}Y_{21} + Y_{22} \\
& \quad + \lambda_1(1 - Y_{11} - Y_{21}) + \lambda_2(1 - Y_{12} - Y_{22}) \\
& \quad + \pi_1(2X_1 + 2X_2 + 2Y_{11} + 3Y_{21} - 4.04) \\
& \quad + \pi_2(2X_1 + 2X_2 + 3Y_{12} + 2Y_{22} - 4.04) \qquad (4.39) \\
& \text{subject to} \quad Y_{11} \leq X_1 \qquad (4.40) \\
& \quad Y_{21} \leq X_2 \qquad (4.41) \\
& \quad Y_{12} \leq X_1 \qquad (4.42) \\
& \quad Y_{22} \leq X_2 \qquad (4.43) \\
& \quad X, Y \in \{0, 1\} \qquad (4.44)
\end{aligned}$$

Rewrite the objective function as

$$\underbrace{(2 + 2\pi_1 + 2\pi_2)}_{\equiv \tilde{f}_1} X_1 + \underbrace{(2 + 2\pi_1 + 2\pi_2)}_{\equiv \tilde{f}_2} X_2 + \underbrace{(1 - \lambda_1 + 2\pi_1)}_{\equiv \tilde{d}_{11}} Y_{11} + \underbrace{(\tfrac{3}{2} - \lambda_2 + 3\pi_2)}_{\equiv \tilde{d}_{12}} Y_{12}$$
$$+ \underbrace{(\tfrac{3}{2} - \lambda_1 + 3\pi_1)}_{\equiv \tilde{d}_{21}} Y_{21} + \underbrace{(1 - \lambda_2 + 2\pi_2)}_{\equiv \tilde{d}_{22}} Y_{22} + \lambda_1 + \lambda_2 - 4.04\pi_1 - 4.04\pi_2 \qquad (4.45)$$

Now let

$$\lambda_1 = 14\zeta \qquad \lambda_2 = 14\zeta \qquad \pi_1 = 2\zeta \qquad \pi_2 = 4\zeta$$

for some constant ζ. Then

$$\tilde{f}_1 = 2 + 4\zeta + 8\zeta = 2 + 12\zeta$$

$$\tilde{f}_2 = 2 + 4\zeta + 8\zeta = 2 + 12\zeta$$

$$\begin{aligned}
\tilde{d}_{11} &= 1 - 14\zeta + 4\zeta &&= 1 - 10\zeta \\
\tilde{d}_{12} &= \tfrac{3}{2} - 14\zeta + 12\zeta &&= \tfrac{3}{2} - 2\zeta \\
\tilde{d}_{21} &= \tfrac{3}{2} - 14\zeta + 6\zeta &&= \tfrac{3}{2} - 8\zeta \\
\tilde{d}_{22} &= 1 - 14\zeta + 8\zeta &&= 1 - 6\zeta
\end{aligned}$$

Suppose $\zeta > \frac{9}{4}$. Then $\tilde{d}_{js} < 0$ for $j = 1, 2,\ s = 1, 2$ and the benefits for the facilities are

$$\begin{aligned}
\tilde{V}_1 &= \tilde{d}_{11} + \tilde{d}_{12} &&= \tfrac{5}{2} - 12\zeta \\
\tilde{V}_2 &= \tilde{d}_{21} + \tilde{d}_{22} &&= \tfrac{5}{2} - 14\zeta
\end{aligned}$$

$\tilde{V}_1 + \tilde{f}_1 = \frac{9}{2} > 0$, so we set $X_1 = Y_{11} = Y_{12} = 0$. Similarly, $\tilde{V}_2 + \tilde{f}_2 = \frac{9}{2} - 2\zeta < 0$ because $\zeta > \frac{9}{4}$, so we set $X_2 = Y_{21} = Y_{22} = 1$. Therefore the objective value is

$$(\tilde{V}_2 + \tilde{f}_2) + (\lambda_1 + \lambda_2) - (4.04\pi_1 + 4.04\pi_2) = (\tfrac{9}{2} - 2\zeta) + (28\zeta) - (24.24\zeta) = 4.5 + 1.76\zeta$$

By making ζ large, we can make the objective value arbitrarily large, so (p-SLR) is unbounded.

(Note that if we had used, say, 5 instead of 4.04 in the right-hand side of the p-robust constraints, the problem would have been feasible, and the Lagrangian would not have been unbounded because the objective value would decrease as ζ increases.)

4.2.5 Upper Bound

To attempt to find an upper bound, we start with the facilities opened in the lower-bound solution at each iteration and assign retailers to them as described in Section 3.2.2. The resulting solution may not be feasible. If this solution has a lower cost than the best feasible solution found to date (regardless of whether the solution is itself feasible), we attempt to improve it using the retailer re-assignment heuristic described in Section 3.2.2. We also apply a DC exchange heuristic. This heuristic is similar to that described by Daskin, Coullard, and Shen (2002), except that now one must decide under what circumstances one is willing to make a DC swap that will improve the solution in some scenarios but hurt it in others. For example, suppose scenario 1 is p-feasible under the current solution but scenario 2 is p-infeasible. Are we willing to make a DC exchange if it will help scenario 2 but hurt scenario 1? What if it will make scenario 1 p-infeasible? We use the following rule for DC exchanges. A DC exchange may be made provided that all three of the following conditions hold:

- It decreases the overall expected cost *or* it decreases the cost of a p-infeasible scenario
- It does not make any p-feasible scenario p-infeasible
- It does not increase the cost of any p-infeasible scenario

We make several other modifications to the DC exchange method described by Daskin, Coullard, and Shen. Suppose we are considering swapping facility j out of the solution. We only consider replacing it with facility k if k is one of the 8 nearest facilities to j. The reasoning is that profitable swaps usually involve facilities that are close to each other. Also, when we consider swapping facility j out and facility k in, we do not re-assign all of the retailers. Instead, we re-assign all retailers currently assigned to j to their nearest open facility (including k), and we re-assign any retailer to k if k is closer than the retailer's current facility. Note that we are making these assignments based on distance only, not based on inventory savings. Finally, rather than executing the DC exchange heuristic every time a new feasible solution is found, we only execute it every 10 times we find a solution whose objective value is 1.2UB or less, where UB is the cost of the best feasible solution found at the current node; the DC exchange heuristic is also performed at the end of the Lagrangian procedure at each node. (The size of the "neighborhood" considered for swapping (8), the threshold value (1.2), and the "frequency" (every 10 iterations) are parameters of the algorithm that can be easily adjusted. In general, increasing the neighborhood size, threshold, and frequency results in higher-quality solutions and longer run times.)

4.2.6 Branch and Bound

If the bounds returned by the Lagrangian relaxation procedure are larger than the desired optimality gap, or if no feasible solution has been found and the lower bound is not greater than Q, then we use branch-and-bound as described in Section 3.2.3. The branch-and-bound procedure may terminate with either a feasible solution having been found or none having been found. If one has been found and the lower and upper bounds from the branch-and-bound tree are within the desired tolerance, then the algorithm terminates; an optimal solution has been found. If a feasible solution has been found but the optimality gap is too large, we must branch on the assignment (Y) variables to close the gap, even if all facilities have been fixed open or closed by the variable-fixing routine (as in the algorithms for the LMRP and the SLMRP). If, on the other hand, no feasible solution has been found when the branch-and-bound procedure terminates, we must examine the best overall lower bound. If this lower bound is greater than Q, we can stop and claim that the problem is infeasible. But if the lower bound is not greater than Q, we cannot conclude whether the problem is feasible or infeasible, and we must again branch on the Y variables to resolve the issue.

As in the previous algorithms, the variable chosen for branching is the unfixed facility with the largest assigned demand in the best feasible solution found at the current node. If no feasible solution has been found at the current node but a feasible solution has been found elsewhere in the branch-and-bound tree, that solution is used instead. If no feasible solution has been found anywhere in the tree, the unfixed facility with the largest expected demand (of the retailer located at that facility) is chosen for branching.

4.2.7 Variable Fixing

The variable-fixing procedure described in Section 3.2.3 can be used within the branch-and-bound method for the p-SLMRP. However, one can also perform variable fixing in the pre-processing step. Recall that during pre-processing, the values z_s^* must be computed; this entails solving $|S|$ single-scenario LMRP problems. When each problem has been solved, we perform the following test. For a given scenario, let $\tilde{V}_{js}$ be the facility benefits (the optimal objective values of the problems (SP$_{js}$)) under a particular set of Lagrange multipliers λ, and let LB be the lower bound (the objective value of (LR)) under the same λ. Suppose that $X_j = 0$ in the solution to (LR) under λ. If

$$\mathrm{LB} + \tilde{V}_{js} + f_j > (1+p)z_s^*,$$

then the scenario under consideration cannot be p-feasible if candidate site j is open, so we can fix $X_j = 0$. Similarly, if $X_j = 1$ and

$$\mathrm{LB} - (\tilde{V}_j + f_j) > (1+p)z_s^*,$$

then site j must be open in every p-robust solution, so we can fix $X_j = 1$. By performing this check for each facility j and each scenario s, we obtain two lists, one containing facilities that must be closed and the other containing facilities that must be opened. The corresponding variables may be fixed before beginning to solve (p-SLMRP). If any facility is contained in both lists, we can terminate the algorithm and conclude that the problem is infeasible. This variable-fixing routine serves to shrink the solution space, even before the algorithm proper begins processing.

If facility j is fixed closed for one scenario and open for another, the problem is infeasible for the current value of p and any smaller value. We can use a method like the one just described to obtain a lower bound on the smallest value of p for which the problem is feasible. Let $s \in S$ be fixed, λ a given set of multipliers for the deterministic problem for scenario s, LB the objective value of (LR) under λ, and $\tilde{V}_{js}$ the benefits under the same λ. If $X_j = 0$ in the solution to (LR) under λ, let

$$p_0^s(j) = \frac{\mathrm{LB} + (\tilde{V}_{js} + f_j)}{z_s^*} - 1.$$

If $X_j = 1$, let

$$p_1^s(j) = \frac{\mathrm{LB} - (\tilde{V}_{js} + f_j)}{z_s^*} - 1.$$

(Let $p_0^s(j) = 0$ if $X_j = 1$ and $p_1^s(j) = 0$ if $X_j = 0$.) If $p < p_0^s(j)$ then we must have $X_j = 0$ for scenario s to be p-feasible, and if $p < p_1^s(j)$ then we must have $X_j = 1$. For each j, let

$$p_0(j) = \max_{s \in S} \{p_0^s(j)\}$$

$$p_1(j) = \max_{s \in S} \{p_1^s(j)\}.$$

Then for $p < p(j) = \min\{p_0(j), p_1(j)\}$, the problem is infeasible since j must be both open and closed. Therefore, let

$$\hat{p} = \max_{j \in I} \{p(j)\}.$$

For any $p < \hat{p}$, the problem is infeasible, so $\hat{p}$ provides a lower bound on the minimum value of p for which the problem is feasible. The calculations required to find $\hat{p}$ can be done very quickly using values already available. This method gives us a starting point for finding a good p if we find that our chosen p is too small. It also gives us a lower bound for the minimax regret heuristic discussed in the next section.

4.3 The Minimax Regret Problem

For a given optimization problem with random parameters, the minimax regret problem is to find a solution that minimizes the maximum regret across all scenarios or parameter ranges. One can solve the minimax (relative) regret problem for the LMRP heuristically by systematically varying p and solving (p-SLMRP) for each value. (p-SLMRP) does not need to be solved to optimality: the algorithm can terminate as soon as a feasible solution is found for the current p. The smallest value of p for which the problem is feasible is the minimax regret value. If $\theta = 0$, this procedure serves as a heuristic for the minimax regret UFLP.

We have introduced this method as a heuristic, rather than an exact algorithm. For small or large values of p, it is easy to determine whether (p-SLMRP) is feasible, but for intermediate-range values of p, (p-SLMRP) may be infeasible while its continuous relaxation is feasible. As discussed in Section 4.2.3, infeasibility cannot be detected from the Lagrangian method in this case, and may not be detected until a sizable portion of the branch-and-bound tree has been explored.

Our heuristic for solving the minimax regret LMRP returns two values, p_L and p_U; the minimax relative regret is guaranteed to be in the range $(p_L, p_U]$. The heuristic also returns a solution whose maximum regret is p_U. It works by maintaining four values, $p_L \leq \bar{p}_L \leq \bar{p}_U \leq p_U$ (see Figure 4.2). At any point during the execution of the heuristic, the problem is known to be infeasible for $p \leq p_L$ and feasible for $p \geq p_U$; for $p \in [\bar{p}_L, \bar{p}_U]$, the problem is indeterminate (i.e., feasibility has been tested but could not be determined); and for $p \in (p_L, \bar{p}_L)$ or $(\bar{p}_U, p_U)$, feasibility has not been tested. At each iteration, a value of p is chosen in $(p_L, \bar{p}_L)$ or $(\bar{p}_U, p_U)$ (whichever range is larger), progressively reducing these ranges until they are both smaller than some pre-specified tolerance ϵ.

Figure 4.2: Ranges maintained by the minimax-regret heuristic.

Algorithm 4.1 (MINIMAX-REGRET)

0. Determine a lower bound p_L for which (p-SLMRP) is known to be infeasible and an upper bound p_U for which (p-SLMRP) is known to be feasible. Let (X^*, Y^*) be a feasible solution with maximum regret p_U. Mark $\bar{p}_L$ and $\bar{p}_U$ as undefined.

1. If $\bar{p}_L$ and $\bar{p}_U$ are undefined, let $p \leftarrow (p_L + p_U)/2$; else if $\bar{p}_L - p_L > p_U - \bar{p}_U$, let $p \leftarrow (p_L + \bar{p}_L)/2$; else, let $p \leftarrow (\bar{p}_U + p_U)/2$.

2. Determine the feasibility of (p-SLMRP) under the current value of p. If it is feasible, let p^* be the maximum relative regret of the solution found.

   2.1 If (p-SLMRP) is feasible, let $p_U \leftarrow p^*$, let (X^*, Y^*) be the solution found in step 2, and go to step 3.

   2.2 Else if (p-SLMRP) is infeasible, let $p_L \leftarrow p$ and go to step 3.

   2.3 Else [(p-SLMRP) is indeterminate]: If $\bar{p}_L$ and $\bar{p}_U$ are undefined, let $\bar{p}_L \leftarrow p$ and $\bar{p}_U \leftarrow p$ and mark $\bar{p}_L$ an $\bar{p}_U$ as defined; else if $p \in (p_L, \bar{p}_L)$, let $\bar{p}_L \leftarrow p$; else [$p \in (\bar{p}_U, p_U)$], let $\bar{p}_U \leftarrow p$. Go to step 3.

3. If $\bar{p}_L - p_L < \epsilon$ and $p_U - \bar{p}_U < \epsilon$, stop and return $p_L,\ p_U,\ (X^*, Y^*)$. Else, go to step 2.

Several comments are in order. In step 0, the lower bound p_L can be determined either by choosing a small enough value that the problem is known to be infeasible (e.g., 0) or by setting $p_L \leftarrow \hat{p}$ found using the method described in Section 4.2.7. The upper bound p_U can be determined by solving the SLMRP (i.e., setting $p = \infty$) and setting

p_U equal to the maximum regret value from the solution found; this solution can also be used as (X^*, Y^*). In step 1, we are performing a binary search on each region. More efficient line searches, such as the Golden Section search, would work as well, but we use the binary search for ease of exposition. In step 2, the instruction "determine the feasibility..." is to be carried out by solving (p-SLMRP) until (a) a feasible solution has been found [the problem is feasible], (b) the lower bound exceeds the artificial upper bound Q [the problem is infeasible], or (c) a pre-specified stopping criterion has been reached [the problem is indeterminate]. This stopping criterion may be specified as a number of Lagrangian iterations, a number of branch-and-bound nodes, a time limit, or any other criterion desired by the user. In general, if the stopping criterion is more generous (i.e., allows the algorithm to run longer), fewer problems will be indeterminate, and the range $(p_L, p_U]$ returned by the heuristic will be smaller.

4.4 p-Robust Stochastic Location Problems

The Lagrangian subproblem for (p-SLMRP) discussed in Section 4.2.1.1 has the integrality property, and consequently, the (theoretical) Lagrangian bound is equal to the continuous relaxation bound. In this section we discuss p-robust versions of both the P-median problem (PMP) and the UFLP and present a Lagrangian relaxation algorithm whose subproblem does not have the integrality property, and hence provides tighter bounds. This method can be used in step 2 of Algorithm 4.1 to solve the minimax regret PMP or UFLP heuristically.

4.4.1 p-Robust Stochastic PMP

The p-robust stochastic P-median problem (p-SPMP)[1] is the problem of locating P facilities and assigning retailers to them in a multi-scenario environment to minimize the total expected transportation cost to the retailers from their assigned facilities, subject to a constraint requiring the maximum relative regret to be no more than p. This problem can be thought of as a variation of the p-SLMRP in which all costs except the DC–retailer transportation costs d_{ijs} are equal to 0 and a limit is placed on the number of facilities that can be located. The p-SPMP is formulated as follows:

$$(p\text{-SPMP}) \quad \text{minimize} \quad \sum_{s\in S}\sum_{i\in I}\sum_{j\in I} q_s \mu_{is} d_{ijs} Y_{ijs} \tag{4.46}$$

$$\text{subject to} \quad \sum_{j\in I} Y_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{4.47}$$

$$Y_{ijs} \le X_j \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.48}$$

$$\sum_{i\in I}\sum_{j\in I} \mu_{is} d_{ijs} Y_{ijs} \le (1+p) z_s^* \qquad \forall s \in S \tag{4.49}$$

$$\sum_{j\in I} X_j = P \tag{4.50}$$

$$X_j \in \{0,1\} \qquad \forall j \in I \tag{4.51}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.52}$$

We propose a variable-splitting approach to solve (p-SPMP). (See Section 2.4.3 for a description of variable-splitting applied to capacitated facility location problems.) We add a variable W that will be forced equal to Y; by choosing which set of variables is

[1]The reader is cautioned not to confuse lower-case p, the robustness coefficient, with capital P, the number of facilities to locate.

used in each set of constraints, we obtain a formulation that decomposes nicely when the constraints requiring $W = Y$ are relaxed. The variable-splitting formulation of the (p-SPMP) is as follows:

$$(p\text{-SPMP-VS}) \quad \text{minimize} \quad \beta \sum_{s\in S}\sum_{i\in I}\sum_{j\in I} q_s \mu_{is} d_{ijs} Y_{ijs}$$

$$+(1-\beta)\sum_{s\in S}\sum_{i\in I}\sum_{j\in I} q_s \mu_{is} d_{ijs} W_{ijs} \tag{4.53}$$

$$\text{subject to} \quad \sum_{j\in I} W_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{4.54}$$

$$Y_{ijs} \le X_j \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.55}$$

$$\sum_{i\in I}\sum_{j\in I} \mu_{is} d_{ijs} W_{ijs} \le (1+p) z_s^* \qquad \forall s \in S \tag{4.56}$$

$$\sum_{j\in I} X_j = P \tag{4.57}$$

$$W_{ijs} = Y_{ijs} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.58}$$

$$X_j \in \{0,1\} \qquad \forall j \in I \tag{4.59}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.60}$$

$$W_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.61}$$

The parameter $0 \le \beta \le 1$ ensures that both Y and W are included in the objective function; since $Y = W$, the objective function (4.53) is the same as that of (p-SPMP).

To solve (p-SPMP-VS), we relax constraints (4.58) with Lagrange multipliers λ_{ijs}. Note that in this case, λ is unrestricted in sign. For fixed λ, the resulting subproblem decomposes into an XY-problem and a W-problem:

XY-Problem:

$$\text{minimize} \quad \sum_{s\in S}\sum_{i\in I}\sum_{j\in I}(\beta q_s \mu_{is} d_{ijs} - \lambda_{ijs})Y_{ijs} \tag{4.62}$$

$$\text{subject to} \quad Y_{ijs} \le X_j \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.63}$$

$$\sum_{j\in I} X_j = P \tag{4.64}$$

$$X_j \in \{0,1\} \qquad \forall j \in I \tag{4.65}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.66}$$

W-Problem:

$$\text{minimize} \quad \sum_{s\in S}\sum_{i\in I}\sum_{j\in I}\left[(1-\beta)q_s \mu_{is} d_{ijs} + \lambda_{ijs}\right]W_{ijs} \tag{4.67}$$

$$\text{subject to} \quad \sum_{j\in I} W_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{4.68}$$

$$\sum_{i\in I}\sum_{j\in I}\mu_{is} d_{ijs} W_{ijs} \le (1+p)z_s^* \qquad \forall s \in S \tag{4.69}$$

$$W_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.70}$$

To solve the XY-problem, we compute the benefit $\tilde{V}_j$ of opening each facility j:

$$\tilde{V}_j = \sum_{s\in S}\sum_{i\in I}\min\{0, \beta q_s \mu_{is} d_{ijs} - \lambda_{ijs}\}. \tag{4.71}$$

We set $X_j = 1$ for the P facilities with smallest $\tilde{V}_j$ and set $Y_{ijs} = 1$ if $X_j = 1$ and $\beta q_s \mu_{is} d_{ijs} - \lambda_{ijs} < 0$.

The W-problem reduces to $|S|$ instances of the *multiple-choice knapsack problem* (MCKP), an extension of the classical knapsack problem in which the items are partitioned into classes and exactly one item must be chosen from each class. The MCKP does not have the integrality property, making the bound from this relaxation tighter than the bound that would be obtained by relaxing (4.47) and (4.49), as we did in Section 4.2.1.1. We describe the MCKP and some of the algorithms that have been proposed to solve it in Appendix B.

The W-problem can be formulated using the MCKP as follows. For each scenario $s \in S$, there is an instance of the MCKP. Each instance contains $|I|$ classes, each representing a retailer $i \in I$. Each class contains $|I|$ elements, each representing a facility $j \in I$. Item j in class i has objective function coefficient $(1-\beta)q_s\mu_{is}d_{ijs} + \lambda_{ijs}$ and constraint coefficient $\mu_{is}d_{ijs}$. The right-hand side of the knapsack constraint is $(1+p)z_s^*$.

Either the MCKP must be solved to optimality, or, if a heuristic is used, one must be chosen that returns a *lower bound* on the optimal objective value; otherwise, the Lagrangian subproblem cannot be guaranteed to produce a lower bound for the problem at hand. If the problem is solved heuristically, the variables may be set using the heuristic solution, but then the lower bound used in the subgradient optimization method does not match the actual value of the solution to the Lagrangian subproblem. We have found this mismatch to lead to substantial convergence problems. A better method is to use a lower-bound *solution*, not just the lower bound itself, to set the variables. Not all heuristics that return lower bounds also return lower-bound solutions, however, so care must be taken when making decisions about which MCKP algorithm to use and how to set the variables.

Since the MCKP is NP-hard, we have elected to solve it heuristically by terminating the branch-and-bound procedure of Armstrong et al. (1983), described below, when it reaches a 0.1% optimality gap. This method can be modified to keep track not only of the best lower bound at any point in the branch-and-bound tree, but also a solution attaining that bound. These solutions, which are generally fractional, are then used as the values of W in the Lagrangian subproblem.

Once the XY- and W-problems have been solved, the two objectives are added to obtain a lower bound on the objective function (4.46). An upper bound is obtained using the method outlined in Section 4.2.5. The Lagrange multipliers are updated using subgradient optimization; the method is standard, but the implementation is slightly different than in most Lagrangian algorithms for facility location problems since the lower-bound solution may be fractional.

4.4.2 p-Robust Stochastic UFLP

If $\theta = 0$ in the p-SLMRP, one obtains a p-robust version of the UFLP (p-SUFLP). This problem, too, can be solved using variable-splitting, splitting both the Y variables and the X variables (using variables W and Z, respectively). In addition, the location variables X and Z are indexed by scenario, and a constraint forces locations to be the same in different scenarios:

(p-SUFLP-VS)

$$\text{minimize} \quad \beta \left[\sum_{s \in S} \sum_{j \in I} q_s f_j X_{js} + \sum_{s \in S} \sum_{i \in I} \sum_{j \in I} q_s \mu_{is} d_{ijs} Y_{ijs} \right]$$

$$+(1-\beta) \left[\sum_{s \in S} \sum_{j \in I} q_s f_j Z_{js} + \sum_{s \in S} \sum_{i \in I} \sum_{j \in I} q_s \mu_{is} d_{ijs} W_{ijs} \right] \qquad (4.72)$$

$$\text{subject to} \quad \sum_{j\in I} W_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{4.73}$$

$$Y_{ijs} \leq X_{js} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.74}$$

$$X_{js} = X_{jt} \qquad \forall j \in I, \forall s \in S, \forall t \in S \tag{4.75}$$

$$\sum_{j\in I} f_j Z_{js} + \sum_{i\in I}\sum_{j\in I} \mu_{is} d_{ijs} W_{ijs} \leq (1+p) z_s^* \qquad \forall s \in S \tag{4.76}$$

$$Z_{js} = X_{js} \qquad \forall j \in I, \forall s \in S \tag{4.77}$$

$$W_{ijs} = Y_{ijs} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.78}$$

$$X_{js} \in \{0,1\} \qquad \forall j \in I, \forall s \in S \tag{4.79}$$

$$Z_{js} \in \{0,1\} \qquad \forall j \in I, \forall s \in S \tag{4.80}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.81}$$

$$W_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.82}$$

Relaxing constraints (4.77) and (4.78) with multipliers λ and π, respectively, we obtain a Lagrangian subproblem that decomposes into an XY-problem and a ZW-problem:

XY-Problem:

$$\text{minimize} \quad \sum_{s\in S}\sum_{j\in I} (\beta q_s f_j - \pi_{js}) X_{js} + \sum_{s\in S}\sum_{i\in I}\sum_{j\in I} (\beta q_s \mu_{is} d_{ijs} - \lambda_{ijs}) Y_{ijs} \tag{4.83}$$

$$\text{subject to} \quad Y_{ijs} \leq X_{js} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.84}$$

$$X_{js} = X_{jt} \qquad \forall j \in I, \forall s \in S, \forall t \in S \tag{4.85}$$

$$X_{js} \in \{0,1\} \qquad \forall j \in I, \forall s \in S \tag{4.86}$$

$$Y_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.87}$$

ZW-**Problem:**

$$\text{minimize} \quad \sum_{s\in S}\sum_{j\in I}\left[(1-\beta)q_s f_j + \pi_{js}\right] Z_{js} + \sum_{s\in S}\sum_{i\in I}\sum_{j\in I}\left[(1-\beta)q_s\mu_{is}d_{ijs} + \lambda_{ijs}\right] W_{ijs} \tag{4.88}$$

$$\text{subject to} \qquad \sum_{j\in I} W_{ijs} = 1 \qquad \forall i \in I, \forall s \in S \tag{4.89}$$

$$\sum_{j\in I} f_j Z_{js} + \sum_{i\in I}\sum_{j\in I} \mu_{is}d_{ijs}W_{ijs} \le (1+p)z_s^* \qquad \forall s \in S \tag{4.90}$$

$$Z_{js} \in \{0,1\} \qquad \forall j \in I, \forall s \in S \tag{4.91}$$

$$W_{ijs} \in \{0,1\} \qquad \forall i \in I, \forall j \in I, \forall s \in S \tag{4.92}$$

The XY-problem can be solved by computing the benefit of opening facility j:

$$\tilde{V}_j = \sum_{s\in S}(\beta q_s f_j - \pi_{js}) + \sum_{s\in S}\sum_{i\in I}\min\{0, \beta q_s\mu_{is}d_{ijs} - \lambda_{ijs}\}. \tag{4.93}$$

We set $X_{js} = 1$ for all $s \in S$ (or, equivalently, set $X_j = 1$ in the original problem) if $\tilde{V}_j < 0$, or if $\tilde{V}_k \ge 0$ for all k but is smallest for j. We set $Y_{ijs} = 1$ if $X_{js} = 1$ and $\beta q_s\mu_{is}d_{ijs} < 0$.

The ZW-problem reduces to $|S|$ MCKPs, one for each scenario. As in the p-SPMP, there is a class for each retailer i, each containing an item for each facility j, representing the assignments W_{ijs}; these items have objective function coefficient $(1-\beta)q_s\mu_{is}d_{ijs} + \lambda_{ijs}$ and constraint coefficient $\mu_{is}d_{ijs}$. In addition, there is a class for each facility j, representing the location decisions Z_{js}; these classes contain two items each, one with objective function coefficient $(1-\beta)q_s f_j + \pi_{js}$ and constraint coefficient f_j, representing opening the facility, and one with objective function and constraint coefficient equal to 0, representing not opening the facility. The right-hand side of the knapsack constraint equals $(1 + p)z_s^*$.

We note that the p-SUFLP had even greater convergence problems than the p-SPMP did when an upper-bound solution was used to set the variables, rather than a lower-bound solution, as discussed in Section 4.4.1. This makes the selection of an MCKP algorithm a critical issue for this problem.

4.5 Computational Results

4.5.1 p-SLMRP

4.5.1.1 Experimental Design

We tested our algorithm for the p-SLMRP on the 49-node, 5-scenario data set described in Section 3.4.1, using the same five values of β and θ. The initial value of p is set slightly smaller than the maximum regret from the optimal SLMRP solution. Subsequent values are set as follows. If a feasible solution was found for the previous value of p, the new value of p is set slightly lower than the maximum relative regret from the best solution found; otherwise, the previous p is divided by 2. The process is continued until $p < 0.001$. Each problem is solved until a solution is found within 1% of optimality, or the problem is proved infeasible, or 1000 CPU seconds have elapsed. Other algorithm parameters are given in Table 4.3. The retailer re-assignment and DC exchange heuristics were performed as described in Section 4.2.5.

Table 4.3: Parameters for Lagrangian relaxation algorithm: p-SLMRP.

Parameter	Value
Maximum number of iterations at root node	1200
Maximum number of iterations at other nodes	400
Initial value of α	2
Number of non-improving iterations before halving α	20
Minimum value of α	0.00000001
Minimum LB–UB gap	1%
Initial value for λ_{is}	$10\bar{\mu} + 10f_i$

4.5.1.2 Subgradient Optimization Modifications

Our first step was to settle on a good strategy for subgradient optimization. In Section 4.2.1.2, we discussed two modifications to the standard subgradient optimization procedure: dividing the p-robust constraints by a constant ν times z_s^*, and updating the multipliers λ and π using separate step sizes. In this section we report briefly on the effectiveness of these modifications. We tested the 49-node, 5-scenario problem with $\beta = 0.001, \theta = 0.1$ and $\beta = 0.005, \theta = 1$, and with four different values of p. We tested pooling vs. separating the step-size calculations. For pooled step-size calculations, we tested several values of ν. (When the step-size calculations are separate, the difference in orders of magnitudes of the constraint violations is irrelevant, so varying ν has no effect.) The results are summarized in Table 4.4. The first two columns indicate whether the same step size was used for both sets of multipliers ("Same Step" = Y for pooled, N for separated) and the value of the constraint divisor ν (if the constraints are not divided, this column reads "—"). The remaining columns indicate the lower bound attained for each problem after processing at the root node (the column headers give the value of p). For the sake of compactness, the lower bounds have been divided by 1000. The maximum

Table 4.4: Subgradient optimization modifications.

		$\beta = 0.001, \theta = 0.1$				$\beta = 0.005, \theta = 1$			
Same Step	ν	0.05	0.09	0.12	0.15	0.05	0.09	0.12	0.15
Y	—	-24414	-26663	-28349	-30042	-12073	-14025	-15481	-16936
Y	0.01	-330	-1679	-3246	-4824	229	-383	-1345	-2415
Y	0.1	**176**	-138	-855	-1426	**412**	79	-370	-376
Y	1	174	104	-45	-191	354	277	102	-65
Y	10	168	**170**	168	166	347	**351**	346	342
Y	100	168	168	**168**	168	347	350	**350**	**350**
Y	1000	168	168	168	**168**	347	350	350	**350**
Y	10000	168	168	168	**168**	347	350	350	350
N	—	168	170	168	159	347	348	336	333

value in each column is given in boldface; this is the best lower bound found.

The first row of the table represents the standard subgradient optimization method—pooling the step sizes and not dividing the constraints. This method produces terrible bounds. Separating the step size calculations appears to be an effective technique, though not as effective as pooling the step sizes and choosing ν well. For larger values of p, large values of ν are the most effective in the pooled step-size calculation: if the constraints are generally non-binding, it pays to largely ignore them by dividing them by a large constant. On the other hand, when p is small, the p-robustness constraints must be given greater weight, so smaller values of ν are most effective. Unfortunately, no single value of ν is consistently superior for all values of p. Therefore, we have chosen to use the pooled step-size calculation and to set ν as follows. Let $p_{\max}$ be the maximum relative regret from the optimal solution with $p = \infty$ (i.e., the optimal SLMRP solution). If $p \geq 0.75 p_{\max}$, we set $\nu = 100$; else if $p \geq 0.5 p_{\max}$, we set $\nu = 10$; else if $p \geq 0.25 p_{\max}$, we set $\nu = 1$; else, we set $\nu = 0.1$.

4.5.1.3 Algorithm Performance

The algorithm's performance on the test problems is summarized in Table 4.5. Most of the columns are as described in Section 3.4.2; additional columns are as follows:

p The value of the robustness coefficient, p.

Q The value of the artificial upper bound given in formula (4.20).

Proved Opt? Was the optimal solution found (or was the problem proved infeasible)?

Proved Feas? Was the feasibility of the problem conclusively proved?

When "INFEAS" appears in the Overall UB column, it indicates that the algorithm proved that the problem is infeasible; when it appears in the Root UB column, the algorithm proved infeasibility at the root node. When "DDP" appears in the Overall LB or Root LB columns, it indicates that infeasibility was detected during pre-processing by the variable-fixing routine described in Section 4.2.7. When ∞ appears in the Overall UB or Root UB columns, it indicates that the algorithm did not find a feasible solution, nor did it prove that the problem is infeasible. The desired outcome for each problem is a "Y" in the "Proved Opt?" column—this indicates that an optimal solution was found (within 1%), or the problem was proved infeasible. This outcome holds for 47 of the 75 problems tested (63%). For the 28 problems for which optimality (or infeasibility) could not be proved within the 1000-second time limit, the "Proved Feas?" column indicates whether the feasibility of the problem was established. For 7 of the 75 problems (8%),

Table 4.5: p-SLMRP: Upper and lower bounds.

β	θ	p	Overall LB	Overall UB	Overall Gap	Root LB	Root UB	Root Gap	Q	Proved Opt?	Proved Feas?
0.001	0.1	0.187	168,470	170,131	0.99%	168,402	170,835	1.45%	189,762	Y	Y
0.001	0.1	0.153	168,523	170,194	0.99%	168,424	172,270	2.28%	184,413	Y	Y
0.001	0.1	0.153	168,424	169,728	0.77%	168,424	177,403	5.33%	184,387	Y	Y
0.001	0.1	0.151	168,818	170,506	1.00%	168,425	177,217	5.22%	184,072	N	Y
0.001	0.1	0.144	168,430	171,331	1.72%	168,430	177,403	5.33%	182,896	N	Y
0.001	0.1	0.130	167,284	176,627	5.59%	167,284	∞	—	180,720	N	Y
0.001	0.1	0.130	167,307	∞	—	167,307	∞	—	180,663	N	N
0.001	0.1	0.065	173,205	INFEAS	—	173,205	INFEAS	—	170,295	Y	Y
0.001	0.1	0.032	242,266	INFEAS	—	242,266	INFEAS	—	165,111	Y	Y
0.001	0.1	0.016	185,641	INFEAS	—	185,641	INFEAS	—	162,519	Y	Y
0.001	0.1	0.008	DDP	INFEAS	—	DDP	INFEAS	—	161,223	Y	Y
0.001	0.1	0.004	DDP	INFEAS	—	DDP	INFEAS	—	160,575	Y	Y
0.001	0.1	0.002	DDP	INFEAS	—	DDP	INFEAS	—	160,251	Y	Y
0.001	0.1	0.001	DDP	INFEAS	—	DDP	INFEAS	—	160,089	Y	Y
0.005	0.1	0.197	326,927	329,441	0.77%	326,927	329,441	0.77%	372,094	Y	Y
0.005	0.1	0.177	328,180	330,887	0.83%	328,180	330,887	0.83%	365,925	Y	Y
0.005	0.1	0.165	329,065	330,022	0.29%	329,064	335,425	1.93%	362,217	Y	Y
0.005	0.1	0.161	329,070	332,171	0.94%	329,070	333,743	1.42%	360,734	Y	Y
0.005	0.1	0.133	324,677	333,083	2.59%	324,508	338,866	4.42%	352,041	N	Y
0.005	0.1	0.133	325,181	333,659	2.61%	324,511	335,180	3.29%	352,013	N	Y
0.005	0.1	0.127	325,677	334,158	2.60%	325,374	339,342	4.29%	350,220	N	Y
0.005	0.1	0.123	326,774	335,038	2.53%	325,944	340,504	4.47%	348,997	N	Y
0.005	0.1	0.121	326,714	335,538	2.70%	326,119	342,748	5.10%	348,486	N	Y
0.005	0.1	0.119	327,583	335,730	2.49%	326,515	340,656	4.33%	347,764	N	Y
0.005	0.1	0.114	328,631	335,955	2.23%	327,947	341,707	4.20%	346,118	N	Y
0.005	0.1	0.110	328,137	337,623	2.89%	327,775	∞	—	344,969	N	Y
0.005	0.1	0.108	330,870	336,737	1.77%	328,923	∞	—	344,294	N	Y
0.005	0.1	0.104	331,709	337,169	1.65%	328,694	∞	—	343,148	N	Y
0.005	0.1	0.101	332,957	337,343	1.32%	328,995	∞	—	342,346	N	Y
0.005	0.1	0.099	333,468	340,769	2.19%	329,380	∞	—	341,737	N	Y
0.005	0.1	0.099	330,505	∞	—	329,293	∞	—	341,719	N	N
0.005	0.1	0.089	282,090	∞	—	277,574	∞	—	338,479	N	N
0.005	0.1	0.044	396,307	INFEAS	—	396,307	INFEAS	—	324,648	Y	Y
0.005	0.1	0.034	422,734	INFEAS	—	422,734	INFEAS	—	321,384	Y	Y
0.005	0.1	0.017	330,338	INFEAS	—	330,338	INFEAS	—	316,100	Y	Y
0.005	0.1	0.009	372,586	INFEAS	—	372,586	INFEAS	—	313,458	Y	Y
0.005	0.1	0.004	DDP	INFEAS	—	DDP	INFEAS	—	312,137	Y	Y
0.005	0.1	0.002	DDP	INFEAS	—	DDP	INFEAS	—	311,477	Y	Y
0.005	0.1	0.001	DDP	INFEAS	—	DDP	INFEAS	—	311,146	Y	Y
0.005	0.5	0.184	336,974	340,105	0.93%	336,974	340,105	0.93%	379,885	Y	Y
0.005	0.5	0.168	338,253	340,573	0.69%	338,253	340,573	0.69%	374,518	Y	Y
0.005	0.5	0.148	340,184	342,153	0.58%	339,858	347,429	2.23%	368,220	Y	Y
0.005	0.5	0.129	336,284	343,469	2.14%	335,364	343,469	2.42%	362,246	N	Y
0.005	0.5	0.119	342,394	345,656	0.95%	336,961	348,633	3.46%	358,752	Y	Y
0.005	0.5	0.105	341,319	350,391	2.66%	339,633	∞	—	354,280	N	Y
0.005	0.5	0.094	249,785	∞	—	242,277	∞	—	351,024	N	N
0.005	0.5	0.047	375,940	INFEAS	—	375,940	INFEAS	—	335,877	Y	Y
0.005	0.5	0.024	377,956	INFEAS	—	377,956	INFEAS	—	328,304	Y	Y
0.005	0.5	0.012	424,774	INFEAS	—	424,774	INFEAS	—	324,517	Y	Y
0.005	0.5	0.006	419,226	INFEAS	—	419,226	INFEAS	—	322,624	Y	Y
0.005	0.5	0.003	DDP	INFEAS	—	DDP	INFEAS	—	321,677	Y	Y
0.005	0.5	0.001	DDP	INFEAS	—	DDP	INFEAS	—	321,204	Y	Y
0.005	1	0.196	346,773	349,833	0.88%	346,773	349,833	0.88%	395,743	Y	Y
0.005	1	0.181	347,787	349,777	0.57%	347,787	349,777	0.57%	390,579	Y	Y
0.005	1	0.164	346,672	350,120	1.00%	346,672	350,120	1.00%	385,043	Y	Y
0.005	1	0.145	343,085	351,374	2.42%	342,988	351,374	2.45%	378,616	N	Y
0.005	1	0.126	345,548	352,554	2.03%	345,281	358,120	3.72%	372,370	N	Y
0.005	1	0.112	347,191	354,420	2.08%	347,191	366,626	5.60%	367,740	N	Y
0.005	1	0.095	241,748	361,172	49.40%	236,407	361,172	52.78%	362,072	N	Y

(continued on next page)

Table 4.5: p-SLMRP: Upper and lower bounds (cont'd).

β	θ	p	Overall LB	Overall UB	Overall Gap	Root LB	Root UB	Root Gap	Q	Proved Opt?	Proved Feas?
0.005	1	0.084	304,258	∞	—	299,187	∞	—	358,543	N	N
0.005	1	0.042	420,008	INFEAS	—	420,008	INFEAS	—	344,674	Y	Y
0.005	1	0.021	409,763	INFEAS	—	409,763	INFEAS	—	337,740	Y	Y
0.005	1	0.010	438,592	INFEAS	—	438,592	INFEAS	—	334,273	Y	Y
0.005	1	0.005	485,106	INFEAS	—	485,106	INFEAS	—	332,539	Y	Y
0.005	1	0.003	409,536	INFEAS	—	409,536	INFEAS	—	331,672	Y	Y
0.005	1	0.001	392,755	INFEAS	—	392,755	INFEAS	—	331,239	Y	Y
0.005	20	0.095	555,919	556,799	0.16%	555,919	572,861	3.05%	586,245	Y	Y
0.005	20	0.075	555,434	565,753	1.86%	555,414	∞	—	575,540	N	Y
0.005	20	0.065	556,334	∞	—	556,333	∞	—	570,062	N	N
0.005	20	0.032	557,190	INFEAS	—	557,190	INFEAS	—	552,708	Y	Y
0.005	20	0.016	601,997	INFEAS	—	601,997	INFEAS	—	544,031	Y	Y
0.005	20	0.008	581,270	INFEAS	—	581,270	INFEAS	—	539,692	Y	Y
0.005	20	0.004	675,890	INFEAS	—	675,890	INFEAS	—	537,523	Y	Y
0.005	20	0.002	679,004	INFEAS	—	679,004	INFEAS	—	536,438	Y	Y
0.005	20	0.001	DDP	INFEAS	—	DDP	INFEAS	—	535,896	Y	Y

no feasible solution could be found anywhere within the branch-and-bound tree, but the problem could not be proven infeasible.

The optimality gaps are larger for the p-SLMRP than for the SLMRP. In general, the root-node gaps were on the order of 1% for the loosely-constrained problems and 3-4% for the more tightly constrained problems. As a result, a great deal of branching was required, and in many cases the procedure timed out after 1000 seconds, before coming to a successful resolution. (See Table 4.7 for a summary of Lagrangian iterations, branch-and-bound nodes, CPU times, and variable-fixing.) The branch-and-bound process was often only marginally successful in increasing the lower bound over that found at the root node. We believe that the continuous relaxations of problems with moderately tight p provide weak bounds, but that the lower bounds we find at the root node are close to the theoretical bound.

It is worth noting that in most cases in which infeasibility was proved, it was proved *at the root node*, with no branching required, after a reasonably small number of Lagrangian iterations. For very small values of p, the variable-fixing routine was able to prove infeasibility during pre-processing.

One curious feature of these results is that for moderately tight values of p, the lower bound barely increases as p decreases. This makes problems with p values in that range very difficult to solve. We have experimented with small problems and found that in this range, the continuous relaxation bound stays more or less constant as p decreases, until a sharp increase just before the problem becomes infeasible. This means that the Lagrangian bound will be stagnant in the flat range, as well. The insensitivity of the continuous relaxation bound to the tightening of the constraints is an issue worthy of further study.

In Table 4.6, we summarize the performance of the algorithm for each β/θ pair by reporting the values of p_L, $\bar{p}_L$, $\bar{p}_U$, and p_U from the minimax regret heuristic (see Figure 4.2). Note that this table does not contain the results of *running* the minimax-regret heuristic (those results are given in Table 4.9, below); it is simply intended to summarize the ability of the algorithm to determine feasibility for the p values tested. Recall that the problem is known to be infeasible for $p \le p_L$; feasibility was not tested for $p \in (p_L, \bar{p}_L)$ or $p \in (\bar{p}_U, p_U)$; feasibility was tested but could not be determined for $p \in [\bar{p}_L, \bar{p}_U]$; and the problem is known to be feasible for $p \ge p_U$.

Table 4.7 summarizes the performance of the algorithm. The first six columns are self explanatory. The next three columns indicate the number of facilities fixed open by the variable-fixing routine after processing at the root node, the number fixed closed, and the total number fixed, respectively. The last three columns give the same values for the

Table 4.6: p-SLMRP: Performance summary.

β	θ	p_L	$\bar{p}_L$	$\bar{p}_U$	p_U
0.001	0.1	0.065	0.130	0.130	0.130
0.005	0.1	0.044	0.890	0.990	0.990
0.005	0.5	0.047	0.094	0.094	0.105
0.005	1	0.042	0.084	0.084	0.095
0.005	20	0.032	0.065	0.065	0.075

variable-fixing routine during pre-processing.

From the table it is clear that for the more tightly constrained problems, the larger root-node optimality gaps led to more Lagrangian iterations and branch-and-bound nodes, longer run times, and fewer variables fixed by the variable-fixing routines. The pre-processing routine is effective in forcing facilities open and closed for the smaller values of p, and this decreased the number of iterations required to prove infeasibility since the solution space is reduced by eliminating certain variables. For larger values of p the routine has limited success. Nevertheless, because the variable-fixing checks are extremely fast to execute, the pre-processing variable-fixing routine is still worth implementing.

4.5.2 Expected Cost vs. Maximum Regret

The key purpose of the p-SLMRP is to reduce the maximum regret (by the choice of p) with as little increase in expected cost as possible. Figure 4.3 illustrates this tradeoff. Each curve represents a given value of (β, θ) (as indicated by the legend), and each data point represents the best feasible solution found for a given value of p. (The curve for $\beta = 0.005$, $\theta = 20$ is omitted from the chart for scaling reasons and because it contains only

Table 4.7: p-SLMRP algorithm performance.

β	θ	p	# Lag Iter	# BB Nodes	CPU Time (sec)	# Fixed Open Root	# Fixed Closed Root	Total # Fixed Root	# Fixed Open Preproc	# Fixed Closed Preproc	Total # Fixed Preproc
0.001	0.1	0.187	2463	9	117.3	0	11	11	0	0	0
0.001	0.1	0.153	16412	69	681.0	0	9	9	0	0	0
0.001	0.1	0.153	15517	67	772.1	0	0	0	0	0	0
0.001	0.1	0.151	15339	67	739.9	0	0	0	0	0	0
0.001	0.1	0.144	20315	84	1010.4	0	0	0	0	0	0
0.001	0.1	0.130	31319	116	1002.2	0	0	0	0	0	0
0.001	0.1	0.130	35848	148	1006.6	0	0	0	0	0	0
0.001	0.1	0.065	20	1	1.4	0	0	0	0	2	2
0.001	0.1	0.032	19	1	1.4	0	0	0	0	11	11
0.001	0.1	0.016	16	1	1.3	0	0	0	2	30	32
0.001	0.1	0.008	0	0	0.0	0	0	0	2	38	40
0.001	0.1	0.004	0	0	0.0	0	0	0	5	43	48
0.001	0.1	0.002	0	0	0.0	0	0	0	6	43	49
0.001	0.1	0.001	0	0	0.0	0	0	0	6	43	49
0.005	0.1	0.197	62	1	5.8	2	4	6	0	0	0
0.005	0.1	0.177	109	1	12.5	2	4	6	1	0	1
0.005	0.1	0.165	1247	3	55.3	1	0	1	1	0	1
0.005	0.1	0.161	1242	3	52.1	1	2	3	1	0	1
0.005	0.1	0.133	22274	96	1012.6	1	0	1	1	0	1
0.005	0.1	0.133	21755	90	1016.1	1	0	1	1	0	1
0.005	0.1	0.127	21611	91	1004.8	1	0	1	1	0	1
0.005	0.1	0.123	27788	113	1011.3	1	0	1	1	0	1
0.005	0.1	0.121	26611	118	1003.8	1	0	1	1	0	1
0.005	0.1	0.119	25608	115	1007.8	1	0	1	1	0	1
0.005	0.1	0.114	26412	114	1006.8	1	0	1	1	0	1
0.005	0.1	0.110	26147	116	1005.1	0	0	0	1	0	1
0.005	0.1	0.108	26475	119	1000.9	0	0	0	1	0	1
0.005	0.1	0.104	28041	129	1001.2	0	0	0	2	0	2
0.005	0.1	0.101	29832	121	1001.3	0	0	0	2	0	2
0.005	0.1	0.099	26039	115	1009.6	0	0	0	2	0	2
0.005	0.1	0.099	30749	131	1003.3	0	0	0	2	0	2
0.005	0.1	0.089	27986	131	1003.5	0	0	0	2	0	2
0.005	0.1	0.044	12	1	1.2	0	0	0	2	0	2
0.005	0.1	0.034	10	1	0.9	0	0	0	2	0	2
0.005	0.1	0.017	8	1	0.7	0	0	0	12	6	18
0.005	0.1	0.009	8	1	0.7	0	0	0	17	11	28
0.005	0.1	0.004	0	0	0.0	0	0	0	20	16	36
0.005	0.1	0.002	0	0	0.0	0	0	0	23	21	44
0.005	0.1	0.001	0	0	0.0	0	0	0	26	21	47
0.005	0.5	0.184	50	1	6.4	2	4	6	0	0	0
0.005	0.5	0.168	55	1	6.7	2	4	6	0	0	0
0.005	0.5	0.148	2953	11	136.5	1	0	1	1	0	1
0.005	0.5	0.129	21509	91	1005.1	1	1	2	1	0	1
0.005	0.5	0.119	7987	31	371.2	1	0	1	1	0	1
0.005	0.5	0.105	23368	107	1004.3	0	0	0	1	0	1
0.005	0.5	0.094	24624	109	1008.6	0	0	0	1	0	1
0.005	0.5	0.047	12	1	1.3	0	0	0	2	0	2
0.005	0.5	0.024	10	1	0.9	0	0	0	4	2	6
0.005	0.5	0.012	8	1	0.7	0	0	0	13	7	20
0.005	0.5	0.006	8	1	0.7	0	0	0	17	16	33
0.005	0.5	0.003	0	0	0.0	0	0	0	19	20	39
0.005	0.5	0.001	0	0	0.0	0	0	0	21	22	43
0.005	1	0.196	61	1	6.4	1	3	4	0	0	0
0.005	1	0.181	63	1	7.8	2	6	8	0	0	0
0.005	1	0.164	66	1	7.6	1	3	4	0	0	0
0.005	1	0.145	23254	103	1000.3	1	0	1	1	0	1
0.005	1	0.126	21920	96	1003.3	1	0	1	1	0	1
0.005	1	0.112	22830	101	1004.1	1	0	1	1	0	1

(continued on next page)

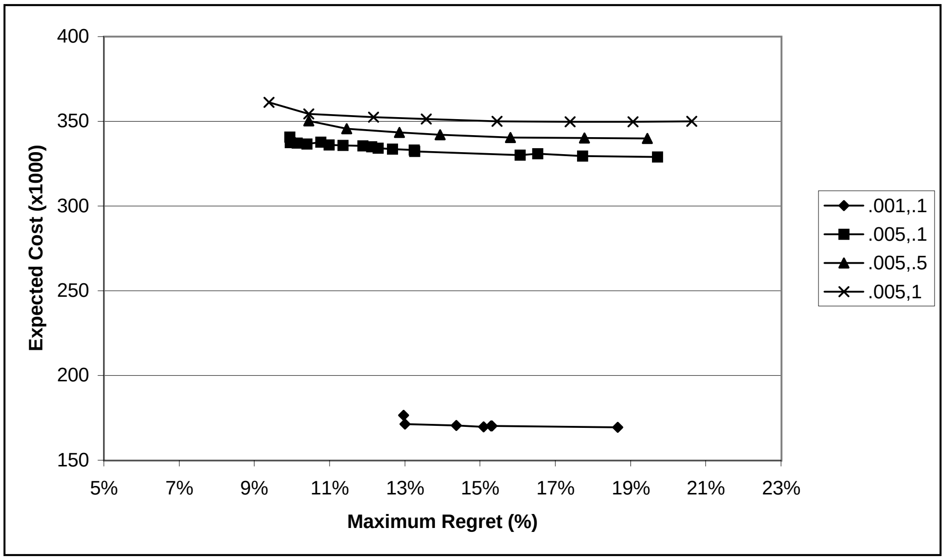
Table 4.7: p-SLMRP algorithm performance (cont'd).

β	θ	p	# Lag Iter	# BB Nodes	CPU Time (sec)	# Fixed Open Root	# Fixed Closed Root	Total # Fixed Root	# Fixed Open Preproc	# Fixed Closed Preproc	Total # Fixed Preproc
0.005	1	0.095	26000	113	1000.0	0	0	0	1	0	1
0.005	1	0.084	27985	122	1007.8	0	0	0	2	0	2
0.005	1	0.042	11	1	0.9	0	0	0	2	0	2
0.005	1	0.021	9	1	0.8	0	0	0	4	5	9
0.005	1	0.010	8	1	0.8	0	0	0	12	6	18
0.005	1	0.005	8	1	0.7	0	0	0	19	12	31
0.005	1	0.003	8	1	0.7	0	0	0	22	17	39
0.005	1	0.001	8	1	0.7	0	0	0	24	19	43
0.005	20	0.095	1232	3	63.8	0	0	0	0	0	0
0.005	20	0.075	23280	99	1011.7	0	0	0	0	0	0
0.005	20	0.065	28638	127	1008.7	0	0	0	1	0	1
0.005	20	0.032	12	1	1.3	0	0	0	2	0	2
0.005	20	0.016	11	1	1.0	0	0	0	3	4	7
0.005	20	0.008	10	1	0.9	0	0	0	6	9	15
0.005	20	0.004	9	1	0.9	0	0	0	9	13	22
0.005	20	0.002	9	1	0.8	0	0	0	13	20	33
0.005	20	0.001	0	0	0.0	0	0	0	15	22	37

three points.) Values of p for which the algorithm found no feasible solution are omitted. The x-axis represents the maximum regret for the solution. The y-axis represents the p-SLMRP objective value. The right-most data point on each curve represents $p = \infty$, i.e., the optimal SLMRP solution.

The figure indicates that in most cases, substantial reductions in maximum regret are possible with small increases in expected cost. For instance, for $\beta = 0.005$, $\theta = 0.1$, the maximum regret can be reduced from 20% to 13% with an increase of only 1% in expected cost; the regret can be further reduced to just under 10% with an increase in cost of less than 4%. The left-most point on the tradeoff curve is not always preferable to decision makers; for example, for $\beta = 0.001$, $\theta = 0.1$ the left-most point represents a decrease in maximum regret from 19% to 13% with an increase in expected cost of 4.3%, but for a slightly larger value of p, a maximum regret of 14% can be obtained with almost no increase in expected cost.

Figure 4.3: Increase in expected cost versus maximum regret.



The data used to generate Figure 4.3 are listed in Table 4.8. The "Expected Cost" column corresponds to the "Overall UB" column in Table 4.8; the remaining columns are described in the explanation of Table 3.6. Rows have been omitted for problems for which no feasible solution was found. Dominated solutions (those for which another solution exists with smaller expected cost and smaller maximum regret) are indicated with an asterisk (*). Had the optimal solution been found for each problem, there would be no dominated solutions; however, since the algorithm used an optimality gap of 0.1% (and did not find solutions within this gap in all cases), some dominated solutions are present. This also means that the true optimal expected cost for some solutions is smaller than that pictured.

Table 4.8: p-SLMRP: Scenario regret.

β	θ	p	% Regret Avg	Min	Max	Expected Cost
0.001	0.1	∞	12.9%	2.3%	18.7%	169,315
0.001	0.1	0.187	12.1%	3.0%	15.3%	170,131*
0.001	0.1	0.153	11.2%	3.8%	15.3%	170,194*
0.001	0.1	0.153	11.5%	2.3%	15.1%	169,728
0.001	0.1	0.151	11.7%	3.4%	14.4%	170,506
0.001	0.1	0.144	10.9%	5.0%	13.0%	171,331
0.001	0.1	0.130	10.9%	6.2%	13.0%	176,627
0.005	0.1	∞	11.6%	0.4%	19.7%	329,042
0.005	0.1	0.197	11.3%	1.0%	17.7%	329,441
0.005	0.1	0.177	11.4%	1.7%	16.5%	330,887*
0.005	0.1	0.165	11.0%	1.9%	16.1%	330,022
0.005	0.1	0.161	9.7%	3.9%	13.3%	332,171
0.005	0.1	0.133	9.1%	4.6%	13.3%	333,083
0.005	0.1	0.133	9.3%	5.0%	12.7%	333,659
0.005	0.1	0.127	9.8%	5.7%	12.3%	334,158
0.005	0.1	0.123	9.2%	5.4%	12.1%	335,038
0.005	0.1	0.121	8.7%	5.7%	11.9%	335,538
0.005	0.1	0.119	9.1%	5.2%	11.4%	335,730
0.005	0.1	0.114	9.2%	6.8%	11.0%	335,955
0.005	0.1	0.110	8.4%	6.4%	10.8%	337,623*
0.005	0.1	0.108	8.5%	7.0%	10.4%	336,737
0.005	0.1	0.104	8.4%	6.3%	10.1%	337,169
0.005	0.1	0.101	8.6%	7.1%	9.9%	337,343
0.005	0.1	0.099	8.8%	6.4%	9.9%	340,769
0.005	0.5	∞	11.4%	0.6%	19.4%	339,959
0.005	0.5	0.184	10.7%	1.4%	17.8%	340,105
0.005	0.5	0.168	10.8%	2.0%	15.8%	340,573
0.005	0.5	0.148	9.4%	3.9%	13.9%	342,153
0.005	0.5	0.129	9.0%	3.7%	12.9%	343,469
0.005	0.5	0.119	8.6%	5.9%	11.5%	345,656
0.005	0.5	0.105	8.9%	6.1%	10.4%	350,391
0.005	1	∞	11.2%	1.0%	20.6%	350,000*
0.005	1	0.196	11.3%	0.3%	19.1%	349,833*
0.005	1	0.181	10.5%	0.9%	17.4%	349,777
0.005	1	0.164	10.6%	1.5%	15.5%	350,120
0.005	1	0.145	9.1%	3.3%	13.6%	351,374
0.005	1	0.126	8.3%	3.1%	12.2%	352,554
0.005	1	0.112	8.3%	5.4%	10.5%	354,420
0.005	1	0.095	8.1%	5.0%	9.4%	361,172
0.005	20	∞	6.6%	1.9%	8.9%	556,665
0.005	20	0.095	6.2%	2.1%	8.5%	556,799
0.005	20	0.075	5.5%	3.7%	7.5%	565,753

Table 4.9: p-SLMRP minimax regret heuristic.

β	θ	p_L	p_U	# Solved
0.001	0.1	7.0%	13.7%	9
0.005	0.1	7.4%	10.7%	9
0.005	0.5	7.3%	10.5%	9
0.005	1	7.0%	10.0%	9
0.005	20	4.9%	7.6%	8

4.5.2.1 Minimax Regret Heuristic

In this section we discuss our testing of the minimax regret heuristic described in Section 4.3. We tested this heuristic on the 49-node, 5-scenario problem, using the same five values of β and θ. No branching was performed, and an iteration limit of 1200 was used (this represents the stopping criteria in step 2 of the heuristic). The results are reported in Table 4.9. The columns marked "p_L" and "p_U" indicate the lower and upper bounds on the minimax regret value; the column marked "# Solved" indicates the total number of problems that were solved during the execution of the algorithm.

4.5.3 p-SPMP and p-SUFLP

4.5.3.1 Algorithm Performance

We tested the variable-splitting algorithms for the p-SPMP and p-SUFLP described in Section 4.4 on two data sets.[2] The first is a 25-node, 5-scenario data set consisting of random data. In scenario 1, demands are drawn uniformly from $[0, 10000]$ and rounded

[2]Although the algorithms proposed in Section 4.4 use Lagrangian relaxation, we will refer to these as the "variable-splitting" algorithms and the algorithm for the p-SLMRP described in Section 4.2 as the "Lagrangian relaxation" algorithm to avoid confusion between the two.

to the nearest integer and latitudes and longitudes are drawn uniformly from [0,1]; in scenarios 2–5, demands from scenario 1 are multiplied by a number drawn uniformly from $[0.5, 1.5]$ and latitudes and longitudes are multiplied by a number drawn uniformly from $[0.75, 1.25]$ (that is, scenario 1 demands are perturbed by up to 50% in either direction, coordinates by up to 25%). Transportation costs are set equal to the Euclidean distances between facilities and customers. Fixed costs for the p-SUFLP problems are drawn uniformly from $[4000, 8000]$ and rounded to the nearest integer. The second data set is the 49-node, 9-scenario data set described in Section 4.5.1.1.

The performance measure of interest for these tests is the tightness of the bounds produced at the root node; consequently, no branching was performed. The parameters used for the variable-splitting algorithm are the same as those used in testing the p-SLMRP algorithm (listed in Table 4.3), except that the minimum LB–UB gap was set to 0.1% and the initial value for all Lagrange multipliers is 0. The weighting coefficient γ was set to 0.2. Values were chosen for the robustness coefficient p using a method similar to that described in Section 4.5.1.1.

Tables 4.10 and 4.11 summarize the p-SPMP algorithm's performance on the 25- and 49-node data sets, respectively. The column marked "P" gives the number of facilities to be located while "p" gives the robustness coefficient. "LB," "UB," and "Gap" give the lower bound, upper bound, and percentage gap after processing at the root node. "# Lag Iter" gives the number of Lagrangian iterations performed, "CPU Time" gives the time (in seconds) spent by the algorithm, and "MCKP Time" gives the time (in seconds) spent solving multiple-choice knapsack problems. Tables 4.12 and 4.13 summarize the

p-SUFLP's algorithm's performance for the 25- and 49-node data sets. The columns are the same as those for tables 4.10 and 4.11, except that the "P" column is not present. As above, "INFEAS" in the UB column indicates that the problem was proved infeasible, while ∞ indicates that the problem was not proved infeasible but no feasible solution was found. Note that since the variable-splitting algorithm cannot be solved solely by the calculation of facility "benefits," variable-fixing cannot be performed, either during pre-processing or after root-node processing. Therefore, no problems can be proved infeasible during pre-processing as in the p-SLMRP algorithm. These tables are summarized in Table 4.14 in a manner similar to Table 4.6.

In general, the bounds are slightly larger than expected. As in the p-SLMRP, some problems could not be proven feasible or infeasible at the root node. Theorem 4.2 implies that for these problems, either the LP relaxation is feasible or we are simply not finding good multipliers. Further research is needed to establish which is the case. Computation times are somewhat longer than for the Lagrangian relaxation algorithm for the p-SLMRP since the subproblems are more difficult to solve; about two-thirds of the total computation time is spent solving MCKPs. Nevertheless, these times are quite reasonable for problems of their size. We discuss these issues further in the next section.

Since the p-SUFLP algorithm requires more variables to be split than the p-SPMP algorithm (the location variables, not just the assignment variables) and requires an additional index on the location variables, we expected this algorithm to produce noticeably weaker bounds. Our results suggest that, to the contrary, the two algorithms produce similarly tight bounds, though more testing would be required to establish this

Table 4.10: p-SPMP algorithm performance: 25-node, 5-scenario data set.

P	p	LB	UB	Gap	# Lag Iter	CPU Time	MCKP Time
3	∞	24,366	24,555	0.8%	1200	20.6	11.4
3	0.156	24,383	24,691	1.3%	1200	22.6	13.4
3	0.072	24,406	∞	—	1200	24.7	15.0
3	0.032	23,947	INFEAS	—	353	7.6	4.7
3	0.012	23,483	INFEAS	—	175	3.7	2.0
3	0.002	23,251	INFEAS	—	137	2.9	1.5
6	∞	13,627	13,640	0.1%	661	11.3	5.7
6	0.201	13,640	13,793	1.1%	1200	42.9	33.1
6	0.177	13,646	14,003	2.6%	1200	22.5	13.1
6	0.169	13,645	13,974	2.4%	1200	22.5	12.6
6	0.151	13,661	∞	—	1200	22.8	12.9
6	0.071	13,507	INFEAS	—	1200	25.9	16.2
6	0.041	13,129	INFEAS	—	127	2.8	1.9
6	0.021	12,876	INFEAS	—	87	1.9	1.3
6	0.011	12,750	INFEAS	—	70	1.5	1.2
6	0.001	12,624	INFEAS	—	61	1.4	1.2
9	∞	8,013	8,036	0.3%	1200	20.7	10.9
9	0.331	8,036	8,460	5.3%	1200	22.8	14.0
9	0.305	8,036	∞	—	1200	23.0	12.4
9	0.145	7,967	INFEAS	—	460	11.7	7.9
9	0.075	7,479	INFEAS	—	88	1.7	1.0
9	0.035	7,201	INFEAS	—	41	0.8	0.6
9	0.015	7,062	INFEAS	—	33	0.6	0.6
9	0.005	6,992	INFEAS	—	31	0.6	0.2
12	∞	4,357	4,361	0.1%	66	1.2	0.7
12	0.403	4,402	4,443	0.9%	1200	29.1	18.9
12	0.309	4,442	4,528	1.9%	1200	32.6	25.6
12	0.232	4,502	∞	—	1200	34.2	25.2
12	0.152	4,511	INFEAS	—	856	31.0	23.6
12	0.072	4,198	INFEAS	—	48	0.9	0.8
12	0.042	4,081	INFEAS	—	34	0.7	0.3
12	0.022	4,002	INFEAS	—	27	0.6	0.4
12	0.012	3,963	INFEAS	—	23	0.5	0.2
12	0.002	3,924	INFEAS	—	20	0.4	0.3
15	∞	2,289	2,291	0.1%	26	0.5	0.2
15	0.450	2,389	2,442	2.2%	1200	28.4	18.8
15	0.409	2,389	2,487	4.1%	1200	27.2	17.9
15	0.323	2,400	∞	—	1200	26.0	16.0
15	0.163	2,423	INFEAS	—	413	8.3	5.7
15	0.083	2,256	INFEAS	—	82	1.5	0.7
15	0.043	2,173	INFEAS	—	37	0.7	0.3
15	0.023	2,131	INFEAS	—	24	0.4	0.2
15	0.013	2,111	INFEAS	—	22	0.4	0.3
15	0.003	2,091	INFEAS	—	20	0.4	0.4

Table 4.11: p-SPMP algorithm performance: 49-node, 9-scenario data set.

P	p	LB	UB	Gap	# Lag Iter	CPU Time	MCKP Time
5	∞	1,376,698	1,408,517	2.3%	1200	93.6	55.3
5	0.212	1,372,735	1,425,534	3.8%	1200	110.3	72.9
5	0.193	1,387,897	∞	—	1200	106.9	69.7
5	0.093	1,382,665	∞	—	1200	140.0	101.0
5	0.043	1,354,732	INFEAS	—	480	54.4	39.3
5	0.023	1,328,748	INFEAS	—	221	26.5	19.9
5	0.013	1,315,756	INFEAS	—	155	20.3	15.1
5	0.003	1,302,765	INFEAS	—	116	15.2	11.7
10	∞	782,946	796,145	1.7%	1200	95.9	57.4
10	0.306	782,664	799,457	2.1%	1200	104.1	65.5
10	0.245	783,093	804,955	2.8%	1200	108.8	69.8
10	0.216	782,901	817,135	4.4%	1200	114.8	76.1
10	0.199	783,003	∞	—	1200	124.6	85.4
10	0.099	783,633	∞	—	1200	134.1	95.4
10	0.049	750,619	INFEAS	—	159	25.2	20.4
10	0.019	729,155	INFEAS	—	77	15.1	12.6
10	0.009	721,998	INFEAS	—	62	12.1	10.6
15	∞	518,381	518,895	0.1%	1125	92.5	57.2
15	0.425	518,885	527,424	1.6%	1200	109.1	71.3
15	0.320	519,294	535,568	3.1%	1200	123.5	85.9
15	0.260	520,377	∞	—	1200	121.5	82.7
15	0.120	518,019	INFEAS	—	313	72.4	62.5
15	0.060	490,276	INFEAS	—	54	27.4	25.5
15	0.030	476,401	INFEAS	—	35	18.8	18.0
15	0.010	467,167	INFEAS	—	28	20.4	19.7
15	0.000	462,593	INFEAS	—	26	13.1	12.2
20	∞	367,573	368,625	0.3%	1200	93.4	56.1
20	0.395	367,973	373,510	1.5%	1200	105.8	68.5
20	0.394	367,915	369,861	0.5%	1200	103.2	65.9
20	0.392	368,005	373,759	1.6%	1200	103.9	66.8
20	0.342	368,157	374,835	1.8%	1200	112.3	73.7
20	0.317	368,650	376,567	2.1%	1200	110.1	72.5
20	0.298	369,153	∞	—	1200	117.8	81.7
20	0.158	353,221	INFEAS	—	103	57.6	54.4
20	0.078	328,816	INFEAS	—	31	13.0	12.1
20	0.038	316,633	INFEAS	—	23	19.9	19.5
20	0.018	310,558	INFEAS	—	21	5.4	4.8
20	0.008	307,515	INFEAS	—	20	7.4	6.7
25	∞	257,815	258,337	0.2%	1200	96.6	57.6
25	0.733	257,862	259,433	0.6%	1200	109.1	69.5
25	0.719	257,668	259,682	0.8%	1200	106.3	68.6
25	0.719	257,689	259,974	0.9%	1200	106.5	69.5
25	0.638	257,825	260,416	1.0%	1200	108.2	70.9
25	0.629	257,852	262,763	1.9%	1200	105.1	67.0
25	0.624	258,147	262,480	1.7%	1200	108.7	71.8
25	0.623	257,929	266,843	3.5%	1200	109.1	72.2
25	0.604	257,941	262,618	1.8%	1200	107.9	67.3
25	0.581	258,103	264,574	2.5%	1200	133.8	96.2
25	0.558	258,517	∞	—	1200	111.8	73.7

(continued on next page)

Table 4.11: p-SPMP algorithm performance: 49-node, 9-scenario data set (cont'd).

P	p	LB	UB	Gap	# Lag Iter	CPU Time	MCKP Time
25	0.538	258,669	264,964	2.4%	1200	110.4	73.2
25	0.518	258,896	261,601	1.0%	1200	114.3	75.3
25	0.490	259,357	277,026	6.8%	1200	133.3	96.1
25	0.452	260,200	∞	—	1200	120.4	82.7
25	0.232	250,523	INFEAS	—	100	18.4	15.3
25	0.152	234,265	INFEAS	—	39	9.2	8.0
25	0.072	218,001	INFEAS	—	22	6.8	6.2
25	0.032	209,871	INFEAS	—	18	7.0	6.5
25	0.012	205,847	INFEAS	—	17	6.4	5.7
25	0.002	203,835	INFEAS	—	16	4.6	3.9

Table 4.12: p-SUFLP algorithm performance: 25-node, 5-scenario data set.

p	LB	UB	Gap	# Lag Iter	CPU Time	MCKP Time
∞	40,162	40,184	0.1%	139	2.8	1.2
0.046	40,183	∞	—	1200	26.4	16.3
0.026	40,189	∞	—	1200	27.2	17.4
0.016	39,864	INFEAS	—	136	3.0	2.0
0.006	39,472	INFEAS	—	101	2.3	1.5

Table 4.13: p-SUFLP algorithm performance: 49-node, 9-scenario data set.

p	LB	UB	Gap	# Lag Iter	CPU Time	MCKP Time
∞	1,487,812	1,494,658	0.5%	1200	105.2	63.8
0.115	1,485,672	∞	—	1200	133.4	94.2
0.055	1,467,460	INFEAS	—	206	26.4	20.3
0.025	1,425,721	INFEAS	—	85	14.2	11.4
0.015	1,411,815	INFEAS	—	70	11.9	9.7
0.005	1,397,908	INFEAS	—	61	10.3	7.4

Table 4.14: p-SPMP and p-SUFLP: Performance summary.

n	P	p_L	$\bar{p}_L$	$\bar{p}_U$	p_U
25	3	0.032	0.072	0.072	0.156
25	6	0.071	0.151	0.151	0.169
25	9	0.145	0.305	0.305	0.331
25	12	0.152	0.232	0.232	0.309
25	15	0.163	0.323	0.323	0.409
25	—	0.016	0.026	0.046	∞
49	5	0.043	0.093	0.193	0.212
49	10	0.049	0.099	0.199	0.216
49	15	0.120	0.260	0.260	0.320
49	20	0.158	0.298	0.298	0.317
49	25	0.232	0.452	0.452	0.490
49	—	0.055	0.115	0.115	∞

observation conclusively.

4.5.3.2 Comparison to p-SLMRP Algorithm

The Lagrangian relaxation algorithm from Section 4.2 can be used to solve both the p-SPMP and the p-SUFLP. To solve the p-SUFLP, one simply sets all costs other than the fixed costs f_j and the transportation costs d_{ijs} to 0. To solve the p-SPMP, the fixed costs are also set to 0, and the decision rule for opening facilities must be modified so that the P facilities with minimum benefit $\tilde{V}_j$ are opened, rather than any facility with $\tilde{V}_j + f_j < 0$. Table 4.15 compares the two algorithms' performance on both the p-SPMP and the p-SUFLP using the 49-node, 9-scenario data set. No branching was performed; the table lists root-node statistics only. The column marked "P" contains the value of P used for the p-SPMP, or "—" in the case of the p-SUFLP. The columns marked "LR LB," "LR UB," "LR Gap," and "LR Time" indicate the lower bound, upper bound, percentage gap, and CPU time (in seconds) for the Lagrangian relaxation algorithm; the remaining columns indicate the corresponding values for the variable-splitting algorithms. The bounds are also pictured graphically in Figures 4.4–4.6.

Neither the Lagrangian relaxation algorithm nor the variable-splitting algorithms consistently provide superior bounds. Given this, the Lagrangian relaxation algorithm seems preferable since it is faster. However, the bounds provided by the variable-splitting algorithms are *theoretically* greater than those provided by the Lagrangian algorithm. The Lagrangian relaxation algorithm requires optimizing over $|I||S| + |S|$ multipliers, while the variable-splitting algorithm requires $|I|^2|S|$ multipliers for the p-SPMP and

Table 4.15: Variable-splitting vs. Lagrangian relaxation.

P	p	LR LB	LR UB	LR Gap	LR Time	VS LB	VS UB	VS Gap	VS Time
5	0.250	1,393,880	1,423,360	2.1%	80.9	1,376,698	1,408,517	2.3%	93.6
5	0.212	1,387,950	1,408,520	1.5%	83.4	1,372,735	1,425,534	3.8%	110.3
5	0.193	1,373,880	1,425,530	3.8%	86.6	1,387,897	∞	—	106.9
5	0.153	1,389,430	∞	—	88.6	1,384,491	∞	—	125.0
5	0.113	1,386,250	∞	—	88.2	1,383,140	∞	—	143.3
5	0.083	1,394,560	∞	—	88.4	1,379,001	∞	—	137.7
5	0.063	1,380,880	INFEAS	—	7.5	1,373,177	∞	—	134.2
5	0.043	1,355,020	INFEAS	—	5.4	1,354,732	INFEAS	—	54.4
5	0.023	1,333,280	INFEAS	—	5.4	1,328,748	INFEAS	—	26.5
15	0.450	518,383	518,895	0.1%	15.7	518,381	518,895	0.1%	92.5
15	0.425	518,903	524,564	1.1%	103.2	518,885	527,424	1.6%	109.1
15	0.320	520,425	∞	—	104.7	519,294	535,568	3.1%	123.5
15	0.260	521,029	543,389	4.3%	104.9	520,377	∞	—	121.5
15	0.220	518,145	∞	—	104.9	521,189	∞	—	135.9
15	0.180	523,868	∞	—	105.7	522,949	∞	—	154.9
15	0.140	525,697	∞	—	105.5	527,150	∞	—	140.0
15	0.100	508,810	INFEAS	—	4.5	508,771	INFEAS	—	36.2
15	0.070	500,658	INFEAS	—	5.1	494,898	INFEAS	—	43.3
15	0.050	492,269	INFEAS	—	4.3	485,653	INFEAS	—	37.8
15	0.030	483,885	INFEAS	—	3.0	476,401	INFEAS	—	18.8
15	0.010	468,893	INFEAS	—	2.6	467,167	INFEAS	—	20.4
—	0.150	1,488,770	1,526,210	2.5%	97.2	1,487,812	1,494,658	0.5%	105.2
—	0.115	1,473,440	∞	—	107.6	1,485,672	∞	—	133.4
—	0.095	1,482,370	∞	—	106.4	1,488,636	∞	—	134.7
—	0.085	1,483,470	∞	—	106.4	1,490,346	∞	—	134.0
—	0.075	1,487,430	∞	—	105.6	1,490,687	∞	—	136.7
—	0.065	1,481,360	∞	—	14.2	1,481,373	INFEAS	—	49.4
—	0.055	1,467,590	INFEAS	—	8.4	1,467,460	INFEAS	—	26.4
—	0.045	1,453,840	INFEAS	—	4.9	1,453,548	INFEAS	—	16.6
—	0.035	1,439,790	INFEAS	—	9.2	1,439,634	INFEAS	—	14.7
—	0.025	1,429,840	INFEAS	—	8.9	1,425,721	INFEAS	—	14.2
—	0.015	1,425,980	INFEAS	—	4.7	1,411,815	INFEAS	—	11.9
—	0.005	1,409,380	INFEAS	—	3.8	1,397,908	INFEAS	—	10.3

$|I|^2|S| + |I||S|$ for the p-SUFLP. We believe the larger number of Lagrange multipliers is the reason for the inferior bounds produced by the variable-splitting algorithms *in practice*, and that these bounds can be tightened by improving the multiplier-updating routine. Because the subproblems do not have the integrality property, the variable-splitting algorithms have the potential to produce tighter bounds than the Lagrangian relaxation algorithm. Even though we are not solving the MCKPs to integer optimality, we should still observe bounds tighter than the LP bound since we solve the problems to something greater than LP optimality. That is, we stop the MCKP algorithm with a 0.1%-optimal solution, which typically has a larger objective value than the LP relaxation of the MCKP but smaller than the IP. One problem with our choice of MCKP algorithm is that the data must be converted to non-negative data by adding a positive constant to each objective function coefficient. A 0.1%-optimal solution found by the algorithm may no longer be 0.1%-optimal after the data are converted back into their original form. It may be possible to modify Armstrong et al.'s algorithm to avoid this transformation, or another algorithm may be used that does not require non-negative data. Either way, improving the algorithm to obtain better bounds in practice is a topic for future research.

4.5.3.3 Minimax Regret Heuristic

We solved the minimax regret p-SPMP and p-SUFLP using the minimax regret heuristic discussed in Section 4.3, substituting the variable-splitting algorithm for the Lagrangian relaxation algorithm in Step 2. We tested the heuristic on the 25-node and 49-node data sets described above. The results are summarized in Table 4.16. The first three

Figure 4.4: Variable-splitting vs. Lagrangian relaxation: p-SPMP, $P = 5$.

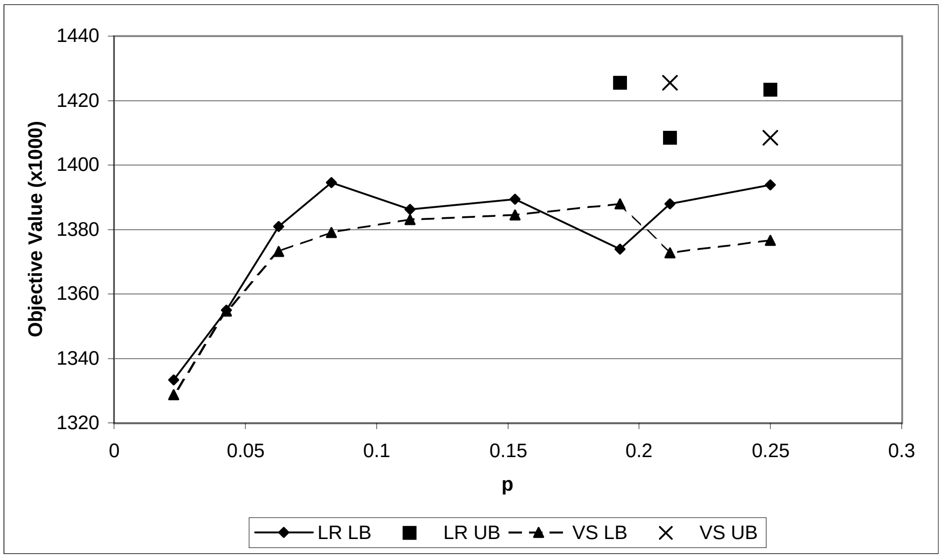


Figure 4.5: Variable-splitting vs. Lagrangian relaxation: p-SPMP, $P = 15$.

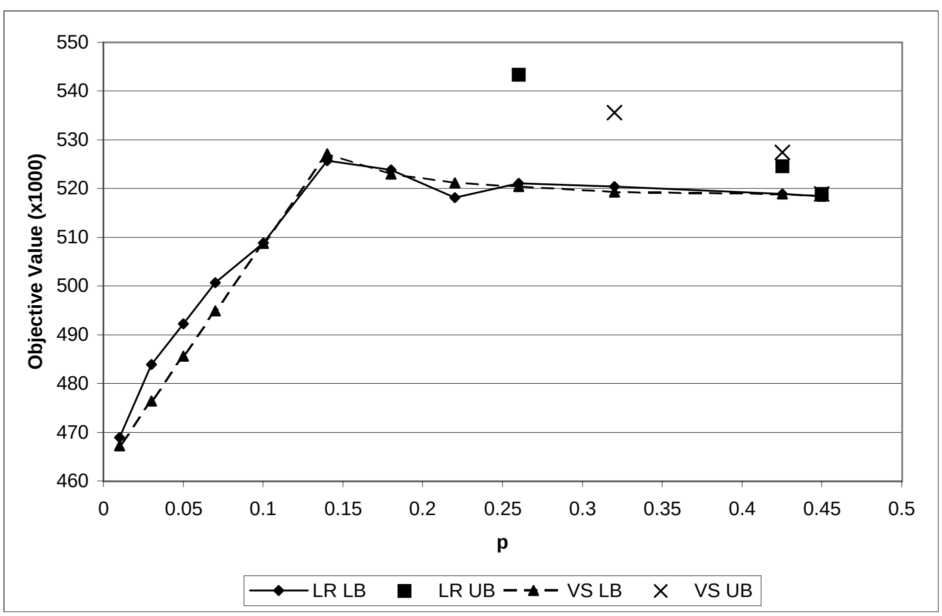


Figure 4.6: Variable-splitting vs. Lagrangian relaxation: p-SUFLP.

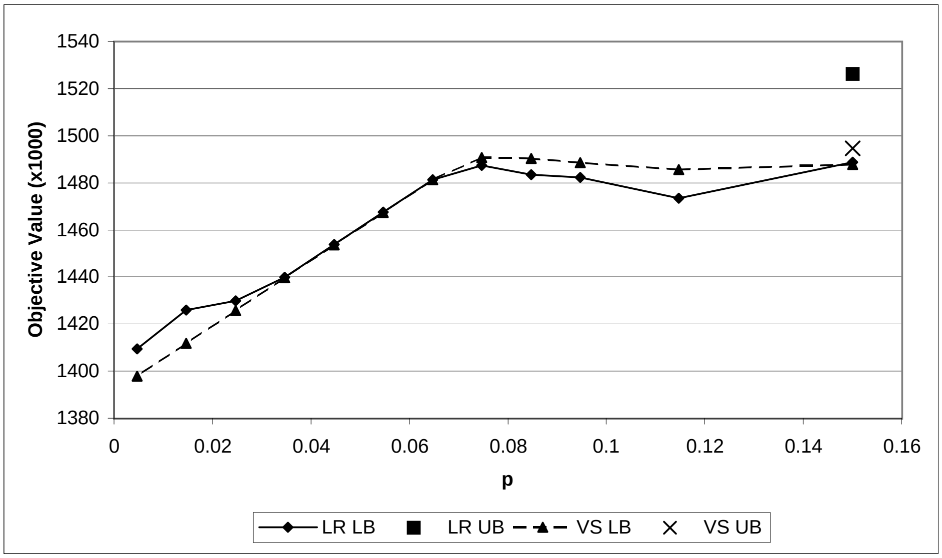


Table 4.16: p-SPMP and p-SUFLP minimax regret heuristic.

# Ret	# Scen	P	p_L	p_U	# Solved
25	5	5	0.040	0.092	8
25	5	10	0.168	0.336	11
25	5	UFLP	0.023	0.046	5
49	9	10	0.087	0.199	10
49	9	20	0.228	0.298	10
49	9	UFLP	0.072	0.115	7

columns indicate the number of retailers and scenarios, and the value of P (if applicable). The remaining three columns are as described in Table 4.9. The ranges returned by the heuristic are smaller for the p-SPMP and p-SUFLP in an absolute sense than those displayed in Table 4.9 for the p-SLMRP (usually less than 1% for the p-SPMP and p-SUFLP and 3%-6% for the p-SLMRP), but in a relative sense (i.e., taking $(p_U - p_L)/p_L$ as opposed to $p_U - p_L$), they are comparable.

4.6 Chapter Summary

In this chapter, we presented the p-robust stochastic LMRP, which finds the minimum-expected-cost p-robust solution to the stochastic LMRP. Our algorithm is similar to the algorithm for the SLMRP but is complicated by the fact that identifying feasible solutions is sometimes difficult, and moreover, it is not always easy to detect in advance whether the problem will be feasible for a given value of p. We presented an upper bound that is valid for any feasible problem; this upper bound can be used during the solution process to detect whether the problem is infeasible. We also presented a heuristic for solving the minimax-regret LMRP, or, as a special case, the minimax-regret UFLP.

The main disadvantage of our algorithm for the LMRP is that the Lagrangian subproblem has the integrality property, meaning that the resulting bound is no tighter than the LP-relaxation bound, which may be quite loose. We show that the p-robust stochastic P-median problem and the p-robust stochastic uncapacitated fixed-charge location problem can be solved using variable-splitting algorithms whose subproblems do not have the integrality property. The subproblems can be solved using algorithms for the multiple-choice knapsack problem (MCKP); since lower bounds are required, we use solutions to restricted LP relaxations of the MCKP, which may be fractional. Although the variable-splitting subproblems are harder to solve than those for the Lagrangian relaxation algorithm, they yield tighter bounds in theory. Our computational results indicate that the bounds produced in practice are not always tighter than those produced using the Lagrangian relaxation method. Our multiplier updating method may be to blame, and we suggest exploring alternate methods as a topic for future research.

Chapter 5

Reliability Models for Facility Location: Maximum Failure Cost

5.1 Introduction

The P-median problem (PMP), uncapacitated fixed-charge location problem (UFLP), and other classical facility location problems choose facility locations and customer assignments to minimize fixed and/or transportation costs. Once a set of facilities has been constructed, however, one or more of them may from time to time become unavailable—for example, due to inclement weather, labor actions, natural disasters, or changes in ownership. These facility "failures" may result in excessive transportation costs as customers previously served by these facilities must now be served by more distant ones. The models presented in this chapter choose facility locations to minimize day-to-day construction and transportation costs while also hedging against failures within the system. We call the ability of a system to perform well even when parts of the system have failed the "reliability" of the system. Our goal, then, is to choose facility locations that are both inexpensive and reliable.

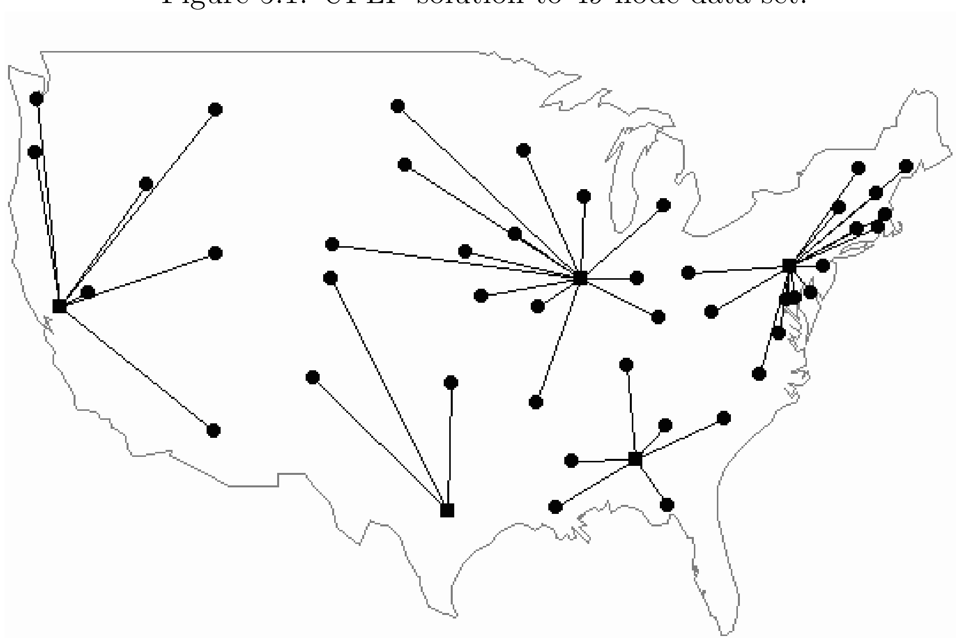
Consider the 49-node data set described in Daskin (1995) consisting of the capitals of the continental United States plus Washington, DC. Demands are proportional to the 1990 state populations and fixed costs to median home values. The optimal UFLP solution for this problem is pictured in Figure 5.1; this solution entails a fixed cost of \$348,000 and a transportation cost of \$509,000. (Transportation costs are taken to be \$0.00001 per mile per unit of demand.) However, if the facility in Sacramento, CA becomes unavailable, the west-coast customers must be served from facilities in Springfield, IL and Austin, TX (Figure 5.2), resulting in a transportation cost of \$1,081,000, an increase of 112%. The "failure costs" (the transportation cost when a site fails) of the five optimal facilities, as well as their assigned demands, are listed in Table 5.1, as is the transportation cost when no facilities fail. Note that Sacramento serves a relatively small portion of the demand; its large failure cost is due to its distance from good "backup" facilities. In contrast, Harrisburg, PA is relatively close to two good backup facilities, but because it serves one-third of the total demand, its failure, too, is very costly. Springfield, IL is the second-largest facility in terms of demand served, but its failure cost is much smaller because it is centrally located, close to good backup facilities. The reliability of a facility, then, can depend either on the distance from other facilities (e.g., Sacramento, which is small but distant) or on the demand served (Harrisburg, which is close but large), or on both (Springfield, which is reliable because it is neither excessively large nor excessively

Table 5.1: Failure costs and assigned demands for UFLP solution.

Location	% Demand Served	Failure Cost	% Increase
Sacramento, CA	19%	1,081,229	112%
Harrisburg, PA	33%	917,332	80%
Springfield, IL	22%	696,947	37%
Montgomery, AL	16%	639,631	26%
Austin, TX	10%	636,858	25%
Transportation cost w/no failures		508,858	0%

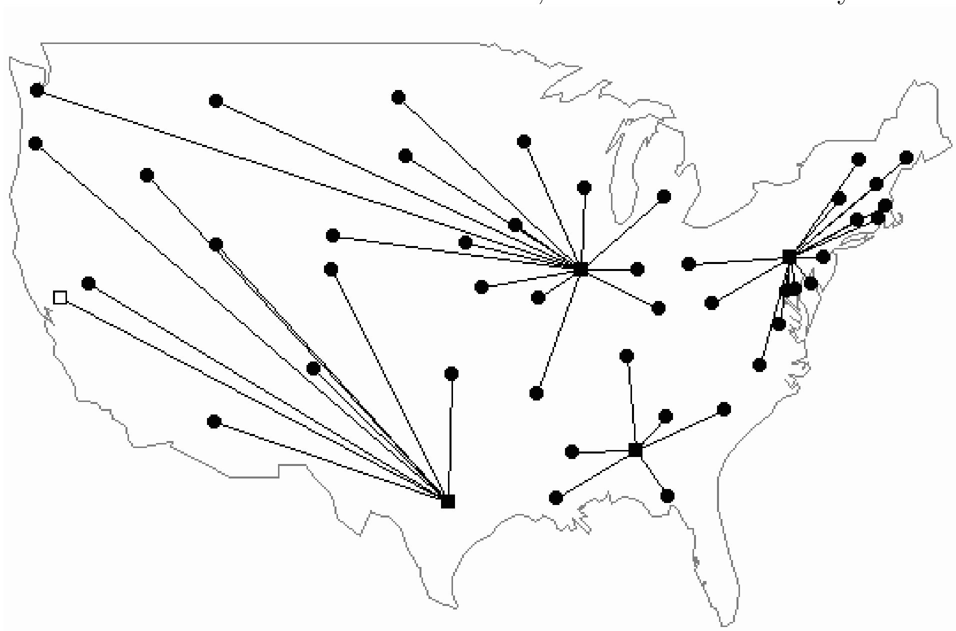
distant).

Figure 5.1: UFLP solution to 49-node data set.



A more reliable solution locates facilities in the capitals of CA, NY, TX, PA, OH, AL, OR, and IA; in this solution, no facility has a failure cost of more than \$640,000, rivaling the smallest failure cost in Table 5.1. On the other hand, three additional facilities are used in this solution, and these come at a cost. Few firms would be willing to choose solutions with fixed and day-to-day transportation costs that are much greater than optimal just to hedge against occasional and unpredictable disruptions in their supply network. One of the goals of this chapter is to demonstrate that substantial improvements

Figure 5.2: UFLP solution to 49-node data set, after failure of facility in Sacramento.



in reliability can often be obtained without large increases in day-to-day operating cost—that by taking reliability into account at design time, one can find a near-optimal UFLP solution that has much better reliability. This is demonstrated by examining the tradeoff between the operating cost and either the *maximum* or the *expected* failure cost of the system.

We study reliability-based formulations of both the PMP and the UFLP. Our ultimate goal is to extend the reliability concept explored in this chapter to the location model with risk pooling (LMRP) discussed in previous chapters. However, the reliability location models are difficult in their own right; we plan to extend them to more complex supply chain models like the LMRP, as well as to other logistics problems, in future research.

In this chapter we discuss models that minimize *operating cost* (fixed location costs and/or day-to-day transportation cost—the classical UFLP or PMP objectives) while constraining the maximum *failure cost* (the transportation cost that results after a facility has failed). We assume that at most one facility can fail at a time and bound the greatest cost that can result from a failure. In the example given in the introduction, the maximum failure cost is 1,081,229, occurring when the facility in Sacramento fails. Chapter 6 will be concerned with the expected failure cost, assuming a given probability that each facility fails.

For the sake of simplicity, we will discuss the reliability problems in detail in the context of the PMP, then briefly discuss how the UFLP-based problems differ. We will refer to the PMP-based version of the maximum failure cost problem as the RPMP-MFC and to the UFLP-based version as the RFLP-MFC. We have developed several formulations for these problems and have explored several approaches for solving them. To date, none of the exact solution methods has been entirely satisfactory. Consequently, we have also developed a tabu search heuristic that executes quickly and finds good solutions in practice.

5.2 Formulations

5.2.1 Notation

Let I be the set of customers, indexed by i, and J be the set of potential facility locations, indexed by j. (We do not assume $I = J$ as we did in earlier chapters to maintain consistency with most of the facility location literature; however, this assumption can easily be made if desired.) The maximum failure cost reliability P-median problem (RPMP-

MFC) is the problem of choosing P facilities from J such that the total transportation cost between customers and their assigned facilities is minimized, subject to a constraint that if any facility fails, the resulting cost after customers are re-assigned is no more than a pre-specified limit. Note that we assume that at most one facility fails at a time. We do not consider either the probability or the duration of a failure; our goal is simply to constrain the cost that results from a failure, regardless of how frequently this cost is incurred.

We define the following notation:

Parameters

h_i = demand per period for customer $i \in I$

d_{ij} = per-unit cost to ship from facility $j \in J$ to customer $i \in I$

P = number of facilities to open, $P \geq 2$

V^* = maximum allowable failure cost

The maximum allowable failure cost V^* may vary from facility to facility (V_j^*) if desired, but we assume for simplicity that the same V^* applies to all facilities. Setting V^* in practice may be a tricky issue, since firms may find it difficult to quantify the maximum failure cost they could tolerate. However, the problem can be solved iteratively with different values of V^* to obtain a tradeoff curve from which decision makers may choose a solution based on their preference between operating cost and failure cost. The method for generating this tradeoff curve is discussed in Section 5.6.

5.2.2 Weak Formulation

The most intuitive formulation for the RPMP-MFC also has the weakest LP relaxation. We present this formulation first, then demonstrate why the LP bound is weak. The weakness of the LP bound also makes Lagrangian relaxation an impractical approach for this formulation.

The basic strategy behind all of the formulations given in this chapter is that each customer is assigned to both a *primary* facility that will serve it under normal circumstances and a *backup* facility that will serve it if the primary facility has failed. (Note that while we refer to a primary or backup "facility," these terms really refer to the assignment, not to the facility itself. A given facility may be a primary facility for one customer and a backup facility for another.) The decision variables are

$$X_j = \begin{cases} 1, & \text{if facility } j \in J \text{ is selected} \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ijk} = \begin{cases} 1, & \text{if facility } j \in J \text{ is customer } i\text{'s primary facility } (i \in I) \text{ and facility } k \in J \text{ is} \\ & \text{customer } i\text{'s backup facility} \\ 0, & \text{otherwise} \end{cases}$$

The weak integer programming formulation of the RPMP-MFC is

$$\text{(RPMP-MFC1)} \quad \text{minimize} \quad \sum_{i \in I} \sum_{j \in J} \sum_{k \in J} h_i d_{ij} Y_{ijk} \tag{5.1}$$

$$\text{subject to} \quad \sum_{j \in J} \sum_{k \in J} Y_{ijk} = 1 \qquad \forall i \in I \tag{5.2}$$

$$Y_{ijk} \leq X_j \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.3}$$

$$Y_{ijk} \leq X_k \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.4}$$

$$\sum_{j \in J} X_j = P \tag{5.5}$$

$$\sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} \leq V^* \qquad \forall j \in J \tag{5.6}$$

$$Y_{ijj} = 0 \qquad \forall i \in I, \forall j \in J \tag{5.7}$$

$$X_j \in \{0, 1\} \qquad \forall j \in J \tag{5.8}$$

$$Y_{ijk} \in \{0, 1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.9}$$

The objective function (5.1) computes the total demand-weighted distance between customers and their primary facilities. (The summation over k is necessary to determine the assignments, but the objective function does not depend on the backup assignments.) Constraints (5.2) require each customer to be assigned to one primary and one backup facility. Constraints (5.3) and (5.4) prevent a customer from being assigned to a primary or a backup facility, respectively, that has not been opened. Constraint (5.5) requires P facilities to be opened.

Constraints (5.6) are the reliability constraints and require the failure cost for facility j to be no greater than V^*. The first summation computes the cost of serving each customer from its primary facility if its primary facility is not j, while the second summation computes the cost of serving customers assigned to j as their primary facility from their backup facilities. Note that this constraint applies to all $j \in J$, not just to those facilities that have been opened. If $X_j = 0$, however, the left-hand side of the constraint reduces to the objective function. Since failure costs are always greater than the P-median cost, this constraint is non-binding if $X_j = 0$.

Constraints (5.7) require a customer's primary facility to be different from its backup facility. (These constraints could have been enforced simply by not defining assignment variables for i, j, k if $j = k$, but we chose to implement them in this way to simplify the variable indexing.) Finally, constraints (5.8) and (5.9) are standard integrality constraints.

If V^* is large, the reliability constraints are non-binding, and solving the RPMP-MFC is equivalent to solving the PMP (the backup assignments are irrelevant in this case), so the RPMP-MFC is NP-hard. This is true of all of the formulations presented in this chapter, including those for the RFLP-MFC.

The following theorem prescribes the values of Y once the X variables are given; it says that a customer will never be assigned to a given facility as a primary facility if a closer facility is open.

Theorem 5.1 *In every optimal solution* (X^*, Y^*) *to (RPMP-MFC1), if* $d_{ij} > d_{ik}$*, then* $Y^*_{ijk} = 0$.

Proof. Suppose (X^*, Y^*) is an optimal solution to (RPMP-MFC1) in which $Y^*_{\hat{\imath}\hat{\jmath}\hat{k}} = 1$

where $d_{\hat{\imath}\hat{\jmath}} > d_{\hat{\imath}\hat{k}}$. Define a new set of assignment variables by

$$\hat{Y}_{ijk} = \begin{cases} 1, & \text{if } i = \hat{\imath},\ j = \hat{k},\ k = \hat{\jmath} \\ 0, & \text{if } i = \hat{\imath},\ j = \hat{\jmath},\ k = \hat{k} \\ Y^*_{ijk}, & \text{otherwise} \end{cases}$$

In other words, $\hat{Y} = Y^*$ except that $\hat{\imath}$'s primary and secondary facilities have been swapped. We show that $(X^*, \hat{Y})$ is a feasible solution to (RPMP-MFC1) with strictly smaller cost.

Clearly

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in J} h_i d_{ij} \hat{Y}_{ijk} < \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} h_i d_{ij} Y^*_{ijk}$$

since $d_{\hat{\imath}\hat{\jmath}} > d_{\hat{\imath}\hat{k}}$, so the cost of the revised solution is smaller. For convenience, define

$$c_j(Y) = \sum_{i\in I}\sum_{\substack{k\in J\\ k\neq j}}\sum_{l\in J} h_i d_{ik} Y_{ikl} + \sum_{i\in I}\sum_{k\in J} h_i d_{ik} Y_{ijk},$$

i.e., $c_j(Y)$ is the left-hand side of constraint (5.6) for facility j under a given set of assignment variables Y.

First consider $j \in J \setminus \{\hat{\jmath}, \hat{k}\}$: $c_j(\hat{Y}) = c_j(Y^*) - h_{\hat{\imath}} d_{\hat{\imath}\hat{\jmath}} + h_{\hat{\imath}} d_{\hat{\imath}\hat{k}}$. Since $d_{\hat{\imath}\hat{\jmath}} > d_{\hat{\imath}\hat{k}}$, $c_j(\hat{Y}) < c_j(Y^*)$, and since (X^*, Y^*) is feasible, $c_j(Y^*) \leq V^*$. Therefore $c_j(\hat{Y}) < V^*$.

Now consider $j = \hat{\jmath}$: $c_j(\hat{Y}) = c_j(Y^*) - h_{\hat{\imath}} d_{\hat{\imath}\hat{k}} + h_{\hat{\imath}} d_{\hat{\imath}\hat{k}} = c_j(Y^*) \leq V^*$. Under $\hat{Y}$, $\hat{k}$ is $\hat{\imath}$'s primary facility instead of its backup, but either way $\hat{\jmath}$'s failure cost includes $h_{\hat{\imath}} d_{\hat{\imath}\hat{k}}$ since $\hat{\imath}$ will be assigned to $\hat{k}$ if $\hat{\jmath}$ fails.

Finally, consider $j = \hat{k}$: $c_j(\hat{Y}) = c_j(Y^*) - h_{\hat{\imath}} d_{\hat{\imath}\hat{\jmath}} + h_{\hat{\imath}} d_{\hat{\imath}\hat{\jmath}} = c_j(Y^*) \leq V^*$, by the same reasoning as for $j = \hat{\jmath}$. Therefore, for all j, $c_j(\hat{Y}) \leq V^*$, as desired. □

Theorem 5.1 implies that once the X variables are known, the Y variables can be set by assigning each customer to its nearest open facility as its primary facility and to its second-nearest open facility as its backup facility. (The optimality of assigning each customer's nearest open facility as its primary facility is evident since the backup assignments do not appear in the objective function.) A similar result applies to all of the formulations presented in this chapter.

5.2.2.1 LP Relaxation of Weak Formulation

The LP relaxation of (RPMP-MFC1), denoted ($\overline{\text{PMP-MFC1}}$), provides a terrible bound on the IP objective value. In fact, in the case in which $I = J$ and the distance between each customer and itself is 0 (a typical setup for location problems), for most values of V^*, the LP relaxation has an objective value of 0:

Theorem 5.2 *Suppose that $I = J$, $d_{ii} = 0$ for all $i \in I$, and for all $j \in J$,*

$$\frac{1}{N-1} h_j \sum_{k \in J} d_{jk} < V^*, \tag{5.10}$$

where $N = |J|$. Then the optimal objective value of ($\overline{\text{PMP-MFC1}}$) is 0.

Proof. Consider the following solution to ($\overline{\text{PMP-MFC1}}$):

$$X_j = \frac{P}{N} \qquad \text{for all } j \in J$$

$$Y_{ijk} = \begin{cases} \frac{1}{N-1}, & \text{if } i = j \text{ and } j \neq k \\ 0, & \text{otherwise} \end{cases}$$

We first show that (X, Y) is a feasible solution to $(\overline{\text{PMP-MFC1}})$. Constraints (5.2) are satisfied because for each $i \in I$,

$$\sum_{j \in J} \sum_{k \in J} Y_{ijk} = \sum_{\substack{k \in J \\ k \neq i}} Y_{iik} = (N-1)\frac{1}{N-1} = 1.$$

Constraints (5.3) are satisfied because $Y_{ijk} \leq \frac{1}{N-1} < \frac{P}{N} = X_j$. (The reader can easily verify that $\frac{1}{N-1} < \frac{P}{N}$ since $2 \leq P \leq N$.) Constraints (5.4) are similar. Constraints (5.5) and (5.7) are trivially satisfied, as are the linear relaxations of the integrality constraints (5.8) and (5.9).

It remains to show that constraints (5.6) are satisfied. For each j,

$$\begin{aligned} \sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} &= \sum_{\substack{i \in I \\ i \neq j}} \sum_{l \in J} h_i d_{ii} Y_{iil} + \sum_{k \in J} h_j d_{jk} Y_{jjk} \\ &= \frac{1}{N-1} h_j \sum_{k \in J} d_{jk} \\ &< V^* \end{aligned}$$

The first equality follows from the fact that every retailer's primary facility is itself, while the second follows from the fact that $d_{ii} = 0$ for all i and from the definition of Y_{ijk}. The inequality follows from the theorem's assumption. Therefore (X, Y) is feasible. Since $Y_{ijk} > 0$ only if $i = j$ and $d_{ii} = 0$, the objective value of (X, Y) is 0. □

The left-hand side of (5.10) is customer j's demand times the average distance from j to the other customers. In general, this value will be quite small compared to the optimal PMP cost since it is roughly equal to the transportation cost for only a single customer. Since V^* is always greater than the optimal PMP cost, the theorem applies to nearly every reasonable value of V^*.

5.2.3 Strong Formulation

A stronger formulation of the RPMP-MFC can be obtained by replacing the linking constraints (5.3) with the following set of constraints:

$$\sum_{k \in J} Y_{ijk} \leq X_j \qquad \forall i \in I, \forall j \in J.$$

The LP solution given in the proof of Theorem 5.2 is not feasible for the strong formulation, so constraints (5.13) act like a cut, tightening the formulation significantly. The resulting formulation will be referred to as the "strong formulation":

$$\text{(RPMP-MFC2)} \quad \text{minimize} \quad \sum_{i \in I} \sum_{j \in J} \sum_{k \in J} h_i d_{ij} Y_{ijk} \tag{5.11}$$

$$\text{subject to} \quad \sum_{j \in J} \sum_{k \in J} Y_{ijk} = 1 \qquad \forall i \in I \tag{5.12}$$

$$\sum_{k \in J} Y_{ijk} \leq X_j \qquad \forall i \in I, \forall j \in J \tag{5.13}$$

$$Y_{ijk} \leq X_k \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.14}$$

$$\sum_{j \in J} X_j = P \tag{5.15}$$

$$\sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} \leq V^* \qquad \forall j \in J \tag{5.16}$$

$$Y_{ijj} = 0 \qquad \forall i \in J, \forall j \in J \tag{5.17}$$

$$X_j \in \{0, 1\} \qquad \forall j \in I \tag{5.18}$$

$$Y_{ijk} \in \{0, 1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.19}$$

The strong formulation has a much tighter bound, as shown empirically in Section 5.10.1.

5.2.4 Separable Formulation

In this section we present another formulation of the RPMP-MFC whose main advantage is that it lends itself to a Lagrangian relaxation that is separable by facility and whose subproblem does not have the integrality property. In this formulation, called the "separable formulation," the location variables are as in earlier formulations ($X_j = 1$ if facility j is open), but the assignment variables are different. In particular,

$$Y_{ij}^0 = \begin{cases} 1, & \text{if facility } j \text{ is customer } i\text{'s primary facility} \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij}^k = \begin{cases} 1, & \text{if facility } j \text{ serves customer } i \text{ when facility } k \text{ is non-operational} \\ 0, & \text{otherwise} \end{cases}$$

for all $i \in I,\ j,\ k \in J$. In the definition of Y_{ij}^k, "non-operational" means *either* that the facility is open but fails *or* that the facility was not opened in the solution. This is a different interpretation of the assignment variables than is used in previous formulations, since for a given i, $Y_{ij}^k = 1$ for $|J|$ pairs (j,k), whereas in previous formulations, $Y_{ijk} = 1$ for only a single (j,k). The separable formulation is as follows:

$$\text{(RPMP-MFC3)} \quad \text{minimize} \quad \sum_{i\in I}\sum_{j\in J} h_i d_{ij} Y_{ij}^0 \tag{5.20}$$

$$\text{subject to} \quad \sum_{j\in J} Y_{ij}^0 = 1 \qquad \forall i \in I \tag{5.21}$$

$$\sum_{j\in J} Y_{ij}^k = 1 \qquad \forall i \in I, \forall k \in J \tag{5.22}$$

$$Y_{ij}^0 \le X_j \qquad \forall i \in I, \forall j \in J \tag{5.23}$$

$$Y_{ij}^k \le X_j \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.24}$$

$$\sum_{j \in J} X_j = P \tag{5.25}$$

$$Y_{ij}^j = 0 \qquad \forall i \in I, \forall j \in J \tag{5.26}$$

$$\sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij}^k \le V^* \qquad \forall k \in J \tag{5.27}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{5.28}$$

$$Y_{ij}^0 \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{5.29}$$

$$Y_{ij}^k \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.30}$$

The objective function (5.20) sums the fixed costs and the transportation costs between customers and their primary facilities. Constraints (5.21) require each customer to be assigned to a primary facility. Constraints (5.22) require each customer to be assigned to a facility when facility k is non-operational. If k is i's primary facility, constraints (5.22) require i to have a backup facility; otherwise, Y_{ij}^k may be set to 1 for i's primary facility j. We could have formulated (5.22) as $\sum_{j \in J} Y_{ij}^k = X_k$, requiring a backup facility only if k is opened; we chose to formulate these constraints as above to separate X and Y as much as possible, enabling the variable-splitting relaxation presented in Section 5.3.4. Constraints (5.23) and (5.24) prohibit assignments to facilities that are not open. Constraint (5.25) requires P facilities to be opened. Constraints (5.26) require a customer to be served by a facility *other* than j when j is non-operational. Constraints (5.27) are the reliability constraints, requiring the transportation cost when k is not operational to be less than or equal to V^*. Constraints (5.28)–(5.30) are standard integrality constraints.

The LP bounds from all three formulations (weak, strong, and separable) are compared empirically in Section 5.10.1.

5.3 Relaxations

The RPMP-MFC does not lend itself to Lagrangian relaxation as easily as other location models (and their variations) do. For example, in Chapter 4 we solved the p-SLMRP by relaxing the assignment constraints and the p-robustness constraints, which tie the scenarios together. The resulting subproblem decomposes by facility and can be solved by computing the benefit of each. The corresponding relaxation for the RPMP-MFC (using any formulation given above) entails relaxing the assignment constraints and the reliability constraints, but the resulting subproblem is not separable by facility and cannot easily be solved. However, other relaxations are possible. Some of these are discussed next. Except where noted, in all of the relaxations below, upper bounds are obtained by opening the facilities that are open in the solution to the Lagrangian subproblem and assigning customers in order of distance, and multipliers are updated using standard subgradient optimization (or a variation of it similar to that described in Section 4.2.1.2).

The four relaxations discussed below (the LLR relaxation, the ALR relaxation, the hybrid relaxation, and the variable-splitting relaxation) are compared empirically in Section 5.10.2.

5.3.1 LLR Relaxation

Suppose constraints (5.3), (5.4), and (5.6) are relaxed in (RPMP-MFC1). We will refer to this relaxation as the "LLR relaxation" since we are relaxing two sets of **L**inking constraints and the **R**eliability constraints. The resulting subproblem (for given Lagrange multipliers λ, μ, π) is

$$\begin{aligned}
\text{(LLR)} \quad \text{minimize} \quad & \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} h_i d_{ij} Y_{ijk} + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} \lambda_{ijk}(Y_{ijk} - X_j) \\
& + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} \mu_{ijk}(Y_{ijk} - X_k) \\
& + \sum_{j\in J} \pi_j \left(\sum_{i\in I}\sum_{\substack{k\in J \\ k\neq j}}\sum_{l\in J} h_i d_{ik} Y_{ikl} + \sum_{i\in I}\sum_{k\in J} h_i d_{ik} Y_{ijk} - V^* \right) \\
= & \sum_{j\in J} \tilde{f}_j X_j + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} \tilde{d}_{ijk} Y_{ijk} + C
\end{aligned} \tag{5.31}$$

$$\text{subject to} \quad \sum_{j\in J}\sum_{k\in J} Y_{ijk} = 1 \qquad \forall i \in I \tag{5.32}$$

$$\sum_{j\in J} X_j = P \tag{5.33}$$

$$Y_{ijk} = 0 \qquad \forall i \in I, \forall j \in J, \forall k \in J \text{ s.t. } d_{ij} > d_{ik} \tag{5.34}$$

$$Y_{ijj} = 0 \qquad \forall i \in I, \forall j \in J \tag{5.35}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{5.36}$$

$$Y_{ijk} \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.37}$$

In the objective function,

$$\tilde{f}_j = \sum_{i \in I} \sum_{k \in J} -(\lambda_{ijk} + \mu_{ikj})$$

$$\tilde{d}_{ijk} = h_i d_{ij} \left(1 + \sum_{\substack{l \in J \\ l \neq j}} \pi_l \right) + \lambda_{ijk} + \mu_{ijk} + \pi_j h_i d_{ik}$$

$$C = -V^* \sum_{j \in J} \pi_j$$

Constraints (5.34) are not needed in (RPMP-MFC1) by Theorem 5.1. However, solutions to (LLR) may not automatically satisfy (5.34) since the objective function is no longer based solely on distance; thus, adding the constraints tightens the formulation.

This problem decomposes into separate problems for X and Y. To solve the X problem, we set $X_j = 1$ for the P facilities with the smallest value of $\tilde{f}_j$. To solve the Y problem, we set $Y_{ijk} = 1$ for the j, k with the smallest value of $\tilde{d}_{ijk}$, provided that $d_{ij} \leq d_{ik}$ and $j \neq k$.

This relaxation generally yields lower bounds of 0, which should not be surprising since it is based on the weak relaxation, whose LP relaxation generally has bounds of 0, and since the Lagrangian subproblem has the integrality property. The strengthening constraints (5.13) cannot be used in the LLR relaxation since its solution depends on the separability of X and Y.

5.3.2 ALR Relaxation

Now suppose we relax the **A**ssignment constraints (5.12), the second set of **L**inking constraints (5.14), and the **R**eliability constraints (5.16) in (RPMP-MFC2). The resulting subproblem (for given λ, μ, π) is

$$\text{(ALR)} \quad \text{minimize} \quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} h_i d_{ij} Y_{ijk} + \sum_{i\in I} \lambda_i \left(1 - \sum_{j\in J}\sum_{k\in J} Y_{ijk}\right)$$

$$+ \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} \mu_{ijk}(Y_{ijk} - X_k)$$

$$+ \sum_{j\in J} \pi_j \left(\sum_{i\in I} \sum_{\substack{k\in J\\ k\neq j}} \sum_{l\in J} h_i d_{ik} Y_{ikl} + \sum_{i\in I}\sum_{k\in J} h_i d_{ik} Y_{ijk} - V^* \right)$$

$$= \sum_{j\in J} \tilde{f}_j X_j + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} \tilde{d}_{ijk} Y_{ijk} + C \tag{5.38}$$

$$\text{subject to} \quad \sum_{k\in J} Y_{ijk} \le X_j \qquad \forall i \in I, \forall j \in J \tag{5.39}$$

$$\sum_{j\in J} X_j = P \tag{5.40}$$

$$Y_{ijk} = 0 \qquad \forall i \in I, \forall j \in J, \forall k \in J \text{ s.t. } d_{ij} > d_{ik} \tag{5.41}$$

$$Y_{ijj} = 0 \qquad \forall i \in I, \forall j \in J \tag{5.42}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{5.43}$$

$$Y_{ijk} \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.44}$$

In the objective function,

$$\tilde{f}_j = \sum_{i\in I}\sum_{k\in J} -\mu_{ikj}$$

$$\tilde{d}_{ijk} = h_i d_{ij} \left(1 + \sum_{\substack{l\in J\\ l\neq j}} \pi_l \right) - \lambda_i + \mu_{ijk} + \pi_j h_i d_{ik}$$

$$C = \sum_{i\in I} \lambda_i - V^* \sum_{j\in J} \pi_j$$

This subproblem allows a customer to be assigned to a secondary facility that is not open, but not to a primary facility that is not open. Constraints (5.39) dictate that a customer assigned to j as a primary facility may be assigned to at most one backup facility; this will be the backup facility k that minimizes $\tilde{d}_{ijk}$, provided $k \neq j$ and $d_{ij} \leq d_{ik}$. Therefore, the benefit of each facility j is:

$$\gamma_j = \tilde{f}_j + \sum_{i \in I} \min \left\{ 0, \min_{\substack{k \in J \\ k \neq j \\ d_{ij} \leq d_{ik}}} \{\tilde{d}_{ijk}\} \right\}. \tag{5.45}$$

To solve (ALR), we set $X_j = 1$ for the P facilities with the smallest γ_j and set $Y_{ijk} = 1$ if $X_j = 1$ and k attains the inner minimization in (5.45).

5.3.3 Hybrid Relaxation

In this section we discuss a "hybrid" relaxation in which some constraints are relaxed using Lagrangian relaxation and others are relaxed using what we will call "bootstrap" relaxation. The advantage of this relaxation is that the subproblem does not have the integrality property, so it provides a tighter theoretical bound than (ALR).

First, consider the reliability constraints (5.16) in (RPMP-MFC2). We can write the left-hand side

$$\begin{aligned} &\sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} \\ =& \sum_{i \in I} \sum_{k \in J} \sum_{l \in J} h_i d_{ik} Y_{ikl} - \sum_{i \in I} \sum_{l \in J} h_i d_{ij} Y_{ijl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} \end{aligned}$$

$$= \sum_{i \in I} \sum_{k \in J} \sum_{l \in J} h_i d_{ik} Y_{ikl} - \sum_{i \in I} \sum_{k \in J} h_i d_{ij} Y_{ijk} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk}$$

$$= \underbrace{\sum_{i \in I} \sum_{k \in J} \sum_{l \in J} h_i d_{ik} Y_{ikl}}_{=\text{objective function}} + \sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk} \tag{5.46}$$

In other words, the failure cost for facility j is equal to the day-to-day transportation cost (the objective function) plus the difference in cost due to serving customers whose primary facility is j. Now, suppose that $\mathcal{L}$ is a lower bound on the objective function (5.1).

Theorem 5.3

$$\mathcal{L} + \sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk} \leq V^* \tag{5.47}$$

is a relaxation of (5.6).

Proof. It suffices to show that any solution that satisfies (5.2)–(5.9) also satisfies (5.47). Suppose (X, Y) satisfies (5.2)–(5.9). Then

$$\mathcal{L} + \sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk} \leq \sum_{i \in I} \sum_{k \in J} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk}$$

because $\mathcal{L}$ is a lower bound on the objective function, and

$$\sum_{i \in I} \sum_{k \in J} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk} \leq V^*$$

since (X, Y) satisfies (5.6). Therefore (X, Y) satisfies (5.47). □

Our strategy involves replacing (5.6) with (5.47), using the best known lower bound at the current iteration as $\mathcal{L}$, and relaxing the assignment constraints (5.2) and the backup linking constraints (5.4) via Lagrangian relaxation. The reliability constraints (5.6) overlap in the sense that each variable appears in multiple constraints, whereas constraints (5.47) do not overlap; this introduces separability into the problem and allows us to solve it without having to relax (5.6) using Lagrangian relaxation. Each time a new best lower bound is found, $\mathcal{L}$ is updated. The idea is that as $\mathcal{L}$ increases, solutions that were feasible for (5.47) become infeasible, thus increasing the lower bound even further (hence the name "bootstrap" relaxation).

The hybrid relaxation subproblem (for given λ, μ) is as follows:

$$\text{(HR)} \quad \text{minimize} \quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} h_i d_{ij} Y_{ijk} + \sum_{i\in I}\lambda_i\Big(1-\sum_{j\in J}\sum_{k\in J} Y_{ijk}\Big) + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J}\mu_{ijk}(Y_{ijk}-X_k)$$

$$= \sum_{j\in J}\tilde{f}_j X_j + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J}\tilde{d}_{ijk}Y_{ijk} + C \tag{5.48}$$

$$\text{subject to} \quad \sum_{k\in J} Y_{ijk} \le X_j \qquad \forall i\in I, \forall j\in J \tag{5.49}$$

$$\sum_{j\in J} X_j = P \tag{5.50}$$

$$\sum_{i\in I}\sum_{k\in J} h_i d_{ik} Y_{ijk} \le V^* - \mathcal{L} \qquad \forall j\in J \tag{5.51}$$

$$Y_{ijk} = 0 \qquad \forall i\in I, \forall j\in J, \forall k\in J \text{ s.t. } d_{ij} > d_{ik} \tag{5.52}$$

$$Y_{ijj} = 0 \qquad \forall i\in I, \forall j\in J \tag{5.53}$$

$$X_j \in \{0,1\} \qquad \forall j\in J \tag{5.54}$$

$$Y_{ijk} \in \{0,1\} \qquad \forall i\in I, \forall j\in J, \forall k\in J \tag{5.55}$$

In the objective function,

$$\tilde{f}_j = \sum_{i \in I} \sum_{k \in J} -\mu_{ikj}$$

$$\tilde{d}_{ijk} = h_i d_{ij} - \lambda_i + \mu_{ijk}$$

$$C = \sum_{i \in I} \lambda_i$$

Note that we have included constraints (5.52) to tighten the formulation, as described above.

(HR) decomposes by j. For each j, we compute the benefit of opening j by solving

$$(\text{BEN}_j) \qquad \gamma_j = \text{minimize} \qquad \tilde{f}_j + \sum_{i \in I} \sum_{k \in J} \tilde{d}_{ijk} Y_{ijk} \tag{5.56}$$

$$\text{subject to} \qquad \sum_{k \in J} Y_{ijk} \leq 1 \qquad \forall i \in I, \forall j \in J \tag{5.57}$$

$$\sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} \leq V^* - \mathcal{L} \qquad \forall j \in J \tag{5.58}$$

$$Y_{ijk} = 0 \qquad \forall i \in I, \forall j \in J, \forall k \in J \quad \text{s.t. } d_{ij} > d_{ik} \tag{5.59}$$

$$Y_{ijj} = 0 \qquad \forall i \in I, \forall j \in J \tag{5.60}$$

$$X_j \in \{0, 1\} \qquad \forall j \in J \tag{5.61}$$

$$Y_{ijk} \in \{0, 1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.62}$$

The strong linking constraints (5.13) have been written with a right-hand side of 1 in (5.57) since (BEN_j) assumes that $X_j = 1$. For each i, we must decide whether to assign i to j as a primary facility and, if so, which facility k to assign as a backup facility. (Note that k need not be open.) This problem reduces to a multiple-choice knapsack problem

(MCKP; see Appendix B), as follows. There is a class for each i. Each class contains $|J|+1$ items, one for each $k \in J$ and a dummy item that represents not assigning i to j. The item representing $k \in J$ has objective function coefficient $\tilde{d}_{ijk}$ and constraint coefficient $h_i(d_{ik}-d_{ij})$. The dummy item has objective function coefficient and constraint coefficient equal to 0. The knapsack size is $V^* - \mathcal{L}$. If $k = j$ or $d_{ij} > d_{ik}$, we force the variable to 0 in the MCKP (by setting its objective function coefficient to ∞). To solve (HR), we compute γ_j for each j and open the P facilities with the smallest γ_j.

As in the variable-splitting algorithms for the p-SPMP and p-SUFLP (see Section 4.4.1), we solve the MCKPs to 0.1%-optimality and use the (possibly fractional) lower-bound solution to set the values of Y_{ijk}. The lower-bound solution is the solution to a *constrained* linear program (since it is typically found deeper in the branch-and-bound tree than the root node, when some variables are forced to 0), so it provides a tighter lower bound than the LP relaxation of (BEN$_j$) would.

5.3.4 Variable-Splitting Relaxation

In the separable formulation (RPMP-MFC3), no variable appears in more than one reliability constraint (5.27). We propose a variable-splitting approach to solving this problem (see Sections 2.4.3 and 4.4); the Lagrangian relaxation of the variable-splitting formulation separates by facility since the reliability constraints do not overlap. Moreover, the subproblem does not have the integrality property. The variable-splitting formulation is as follows:

$$\text{(RPMP-VS)} \quad \text{minimize} \quad \beta \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij}^0 + (1-\beta) \sum_{i \in I} \sum_{j \in J} h_i d_{ij} W_{ij}^0 \tag{5.63}$$

$$\text{subject to} \quad \sum_{j \in J} Y_{ij}^0 = 1 \qquad \forall i \in I \tag{5.64}$$

$$\sum_{j \in J} Y_{ij}^k = 1 \qquad \forall i \in I, \forall k \in J \tag{5.65}$$

$$W_{ij}^0 \leq X_j \qquad \forall i \in I, \forall j \in J \tag{5.66}$$

$$W_{ij}^k \leq X_j \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.67}$$

$$\sum_{j \in J} X_j = P \tag{5.68}$$

$$Y_{ij}^j = 0 \qquad \forall i \in I, \forall j \in J \tag{5.69}$$

$$W_{ij}^j = 0 \qquad \forall i \in I, \forall j \in J \tag{5.70}$$

$$\sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij}^k \leq V^* \qquad \forall k \in J \tag{5.71}$$

$$W_{ij}^0 = Y_{ij}^0 \qquad \forall i \in I, \forall j \in J \tag{5.72}$$

$$W_{ij}^k = Y_{ij}^k \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.73}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{5.74}$$

$$Y_{ij}^0 \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{5.75}$$

$$Y_{ij}^k \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.76}$$

$$W_{ij}^0 \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{5.77}$$

$$W_{ij}^k \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.78}$$

Note that constraints (5.26) are included in (RPMP-VS) in both their Y form (5.69) and in their W form (5.70). This is not strictly necessary, but it is easy to include them in both subproblems and doing so tightens the formulation. To solve (RPMP-VS), we relax constraints (5.72) and (5.73); the resulting subproblem (for given λ) decomposes into separate problems, one for X and W and one for Y.

***XW*-Problem:**

$$\text{minimize} \quad (1-\beta)\sum_{i\in I}\sum_{j\in J} h_i d_{ij} W_{ij}^0 + \sum_{i\in I}\sum_{j\in J}\lambda_{ij}^0 W_{ij}^0 + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J}\lambda_{ij}^k W_{ij}^k \tag{5.79}$$

$$\text{subject to} \quad W_{ij}^0 \leq X_j \qquad \forall i \in I, \forall j \in J \tag{5.80}$$

$$W_{ij}^k \leq X_j \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.81}$$

$$\sum_{j\in J} X_j = P \tag{5.82}$$

$$W_{ij}^j = 0 \qquad \forall i \in I, \forall j \in J \tag{5.83}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{5.84}$$

$$W_{ij}^0 \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{5.85}$$

$$W_{ij}^k \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.86}$$

***Y*-Problem:**

$$\text{minimize} \quad \beta\sum_{i\in I}\sum_{j\in J} h_i d_{ij} Y_{ij}^0 + \sum_{i\in I}\sum_{j\in J} -\lambda_{ij}^0 Y_{ij}^0 + \sum_{i\in I}\sum_{j\in J}\sum_{k\in J} -\lambda_{ij}^k Y_{ij}^k \tag{5.87}$$

$$\text{subject to} \quad \sum_{j\in J} Y_{ij}^0 = 1 \qquad \forall i \in I \tag{5.88}$$

$$\sum_{j\in J} Y_{ij}^k = 1 \qquad \forall i \in I, \forall k \in J \tag{5.89}$$

$$Y_{ij}^j = 0 \qquad \forall i \in I, \forall j \in J \tag{5.90}$$

$$\sum_{i\in I}\sum_{j\in J} h_i d_{ij} Y_{ij}^k \leq V^* \qquad \forall k \in J \tag{5.91}$$

$$Y_{ij}^0 \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{5.92}$$

$$Y_{ij}^k \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall k \in J \tag{5.93}$$

To solve the XW-problem, we compute the benefit of each facility. If X_j were set to 1, then we would set $W_{ij}^0 = 1$ if $(1-\beta)h_i d_{ij} + \lambda_{ij}^0 < 0$ and, for $k \in J$, $W_{ij}^k = 1$ if $\lambda_{ij}^k < 0$. Therefore, the benefit of opening facility j is

$$\gamma_j = \sum_{i \in I} \left(\min\{0, (1-\beta)h_i d_{ij} + \lambda_{ij}^0\} + \sum_{k \in J} \min\{0, \lambda_{ij}^k\} \right).$$

We set $X_j = 1$ for the P facilities with minimum γ_j, set $W_{ij}^0 = 1$ if $X_j = 1$ and $(1-\beta)h_i d_{ij} + \lambda_{ij}^0 < 0$, and set $W_{ij}^k = 1$ if $\lambda_{ij}^k < 0$.

To solve the Y-problem, first note that the Y_{ij}^0 variables can be set optimally for each i simply by setting $Y_{ij}^0 = 1$ for the j that minimizes $\beta h_i d_{ij} - \lambda_{ij}^0$, since Y_{ij}^0 does not appear in constraints (5.91). The remaining problem decomposes by k. For each $k \in J$, we solve a MCKP (see Appendix B) defined as follows:

- There is a class for each $i \in I$
- The items in each class correspond to facilities $j \in J$
- The objective function coefficient of item j in class i is λ_{ij}^k
- The constraint coefficient of item j in class i is $h_i d_{ij}$
- The knapsack size is V^*

As in the hybrid relaxation, we use the lower-bound solution returned by the MCKP algorithm to set the Y variables.

5.4 Infeasibility Issues

As with the p-SLMRP, it is not always easy to find a feasible solution to the RPMP-MFC if one exists, nor is it easy to determine *a priori* whether a given instance of the problem is feasible. Like the p-SLMRP, however, we can identify an upper bound on the objective value of any feasible solution to the problem. In particular, it is clear from (5.46) that V^* is itself an upper bound on the objective value since the failure cost is always greater than or equal to the operating cost. Therefore, if the lower bound from any of the relaxations discussed in this chapter ever exceeds V^*, the problem is infeasible; also, V^* can be used as the upper bound in the step-size calculation of the subgradient optimization procedure if no feasible solution has been found.

5.5 Tabu Search Heuristic

The relaxations discussed in the preceding sections offer a promising start for finding good optimization-based methods for solving the RPMP-MFC. However, the bounds produced in practice by these relaxations are not sufficiently tight to make them useful for finding optimal solutions. In addition, the relaxations whose solutions involve the MCKP may not be practical for larger problems since the MCKP is itself NP-hard. For these reasons, we have developed a tabu search heuristic that obtains good-quality solutions with reasonable CPU times, though without any guarantee of optimality.

Tabu search (Glover 1986) is a meta-heuristic that can be applied to any combinatorial optimization problem. The heuristic is based on the idea of a "move," a small, local change to the solution. A move is applied at each iteration and may either improve or degrade the solution; the resulting solution may be infeasible. Once a move is made, it becomes "tabu," or prohibited, for a certain number of iterations. These rules are designed to avoid local optima and to give the algorithm a chance to explore a large portion of the solution space.

The structure of our tabu search algorithm is based on that of Rolland, Schilling, and Current (1996) for the P-median problem. Our handling of infeasibilities is modeled on the tabu search algorithm of Gendreau, Laporte, and Seguin (1996) for the stochastic vehicle routing problem.

5.5.1 Moves and Tabu Lists

We define two types of moves for our algorithm, *adds*, which entail opening a facility not currently in the solution, and *drops*, which entail closing a facility currently in the solution. Since the number of facilities in any optimal solution is fixed at P, performing any move to a feasible solution necessarily makes it infeasible. However, infeasibilities are allowed in tabu search and are in fact beneficial as they help diversify the search. As the algorithm progresses, the allowable difference between P and the actual number of facilities varies to encourage or discourage such diversification. Another common move is the *swap* move, which maintains the number of facilities by simultaneously closing one and opening another. Like Rolland, Schilling, and Current, we have opted not to use the swap move as it requires evaluating $O(|J|^2)$ possible moves at each iteration rather than $O(|J|)$.

When a facility is added, it is inserted into the *add-tabu* list; it may not be reinserted until a given number of iterations, called the *tabu tenure*, have elapsed. Similarly, when a facility is dropped, it is inserted into the *drop-tabu* list until the tabu tenure has elapsed. There is one exception to the tabu rule: if performing a tabu move would produce a feasible solution with objective value less than the current best feasible solution, the move is performed even though it is tabu. This is the *aspiration criterion* used commonly in tabu search algorithms. We use a constant tabu tenure of 6 iterations. There are other ways to set the tabu tenure; for example, Rolland, Schilling, and Current set the tenure randomly. We use the constant-tenure method for simplicity of exposition and because it performs well.

Let N be the number of facilities currently open. The algorithm decides whether to perform an add or a drop at each iteration as follows.

- If $N = 2$, add
- Else if $N = |J|$, drop
- Else if $N < P - s$, add
- Else if $N > P + s$, drop
- Else add with probability 0.5 and drop with probability 0.5

The parameter s is a *slack* parameter that allows the number of open facilities to deviate from P. Initially, s is set to 0; it is increased by 1 whenever the algorithm fails to make improvement in a given number of iterations and is reset to 0 whenever a new best solution is found.

5.5.2 Evaluation of Solutions

To evaluate a given add move, each customer is re-assigned to the new facility if it is closer than its current primary facility; if it is farther than its primary facility but closer than its secondary facility, it is assigned to the new facility as a secondary facility. Similarly, for a drop move, all customers assigned to the dropped facility (as either a primary or secondary facility) must be re-assigned to the remaining facilities. In either case, the resulting solution is evaluated by computing the resulting objective value, then adding an *infeasibility penalty* given by

$$\rho \sum_{j \in J} \max \left\{ 0, \sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} - V^* \right\},$$

i.e., a constant times the sum of the infeasibilities with respect to the reliability constraints. The constant ρ is a self-adjusting penalty coefficient that is initially set to 2. Every 10 iterations, ρ is multiplied by $2^{t/5-1}$, where t is the number of infeasible solutions among the last 10 solutions found. If all of them were feasible, ρ is divided by 2 (thus encouraging more infeasibilities), and if all of them were infeasible, ρ is multiplied by 2 (discouraging infeasibilities).

5.5.3 Initialization and Termination

An initial solution is obtained by greedily adding facilities until P facilities are open, at each step adding the facility that improves the objective value by the greatest amount. Failure costs are not considered during this process, so the resulting solution may not be feasible.

Table 5.2: Parameters for tabu search algorithm for RPMP-MFC.

Parameter	Value
Maximum # of iterations (`maxiter`)	$\max\{100, 2\lvert J\rvert\}$
# of consecutive non-improving iterations after which algorithm terminates	`maxiter`/2
Tabu tenure	6
Initial value of s	0
# of consecutive non-improving iterations after which s is increased by 1	25
Initial infeasibility penalty coefficient ρ	2
Frequency of updating ρ	every 10 iterations

The algorithm terminates when `maxiter` iterations have elapsed, where `maxiter` $= \max\{100, 2|J|\}$, or if a feasible solution has been found but `maxiter`/2 consecutive iterations have failed to improve the solution.

5.5.4 Outline of Tabu Search Heuristic

The relevant parameters for the tabu search heuristic are listed in Table 5.2. Most of them are described above. One of the drawbacks of many tabu search heuristics is the excessive number of parameters. We have tried to keep the number of parameters to a minimum to simplify the exposition of the algorithm. Undoubtedly, our algorithm could be improved by increasing the number of levers that can be adjusted. This would significantly complicate the process of fine-tuning the algorithm, though; moreover, our intent is to demonstrate that tabu search can be used effectively to solve the RPMP-MFC, not to present the best possible tabu search algorithm for it.

We now outline the algorithm itself.

Algorithm 5.1 (TABU-RPMPMFC)

0. Initialize: $\texttt{iter} \leftarrow 1, \texttt{bestcost} \leftarrow \infty, s \leftarrow 0, \texttt{additer}_j \leftarrow -\infty\ \forall j \in J, \texttt{dropiter}_j \leftarrow$

$-\infty \; \forall j \in J$. Generate a starting solution greedily.

1. Choose whether to add or drop (see Section 5.5.1).

   1a. If add, select the best facility j to add as described in Section 5.5.2. If $\texttt{additer}_j + \texttt{tabutenure} \geq \texttt{iter}$, the facility is tabu and can only be added if the resulting solution has cost less than `bestcost`.

   1b. If drop, select the best facility j to drop as described in Section 5.5.2. If $\texttt{dropiter}_j + \texttt{tabutenure} \geq \texttt{iter}$, the facility is tabu and can only be dropped if the resulting solution has cost less than `bestcost`.

2. Set $\texttt{iter} \leftarrow \texttt{iter} + 1$. If the solution found in step 1 is feasible (with respect to the number of facilities and the reliability constraints) and has cost less than `bestcost`, then

   2a. Set $\texttt{bestcost} \leftarrow$ the new solution cost, and store the solution, and

   2b. Set $s \leftarrow 0$.

3. If 10 iterations have passed since the last time ρ was updated, set $\rho \leftarrow \rho \cdot 2^{t/5-1}$, where t is the number of infeasible solutions among the last 10 solutions found.

4. If no improvement has been made in the last 25 iterations, set $s \leftarrow s + 1$.

5. If $\texttt{iter} = \texttt{maxiter}$ or a feasible solution has been found but no improvement has been made in the last $\texttt{maxiter}/2$ iterations, STOP. Else, go to step 1.

Computational results for the tabu search heuristic are presented in Section 5.10.3.

5.6 Tradeoff Curves

Regardless of how the RPMP-MFC is solved, it can be used to generate a tradeoff curve between the operating cost and the maximum failure cost. The tradeoff curve represents a set of non-dominated or Pareto optimal solutions: given a solution on the tradeoff curve and another solution, the solution on the curve is better than the other solution in at least one of the two objectives. Decision makers choose a solution from the tradeoff curve based on their level of preference between the objectives.

We use the constraint method of multi-objective programming (Cohon 1978) to generate the tradeoff curve. The constraint method involves first solving the unconstrained problem ($V^* = \infty$), then setting V^* equal to the maximum failure cost from the solution found minus a small amount and re-solving. The process continues until V^* is small enough that the problem is infeasible. Sample tradeoff curves are shown in Section 5.10.4.

5.7 UFLP-Based Problems

Nearly all of the formulations, relaxations, and algorithms for the RPMP-MFC discussed in the preceding sections can be modified easily to formulate the RFLP-MFC. In general, one simply needs to drop the constraints requiring $\sum_{j\in J} X_j = P$ and add $\sum_{j\in J} f_j X_j$ to the objective function, where f_j is the fixed cost to build facility j, amortized to the time units used to express demands. Most of the results discussed above also apply to the UFLP-based problems. For example, the LP relaxation of the weak formulation is still very weak, and Theorem 5.1 still applies. The X-problem from the LLR relaxation for the RFLP-MFC is solved by opening any facility with $\tilde{f}_j < 0$ (rather than for the P facilities with minimum $\tilde{f}_j$); similarly, the ALR and variable-splitting relaxations are solved by opening facility j if $\gamma_j + f_j < 0$. In both cases, at least two facilities must be opened, even if fewer than two facilities have $\gamma_j + f_j < 0$, since at least two facilities must be open in any feasible solution to the RFLP-MFC.

The hybrid relaxation is not immediately applicable to the RFLP-MFC since it requires the left-hand side of the reliability constraints to be written as the objective function plus a term that includes different variables in each constraint. Since the RFLP-MFC objective function includes $\sum_{j \in J} f_j X_j$ but the reliability constraints do not, the constraints cannot be written in the required form. On the other hand, if the fixed costs are included in the failure cost (a possible variation; see Section 5.8 below), the constraints *can* be written in the required form and the hybrid relaxation can be applied.

The tabu search heuristic is also applicable to the RFLP-MFC, except that the slack parameter s is irrelevant since there is no limit on the number of facilities; also, the fixed costs must be accounted for when computing the change in cost when a facility is added or dropped. In addition, in the procedure for finding an initial solution we continue opening facilities greedily if the current solution is infeasible, even if doing so increases the cost. This guarantees that a feasible solution will be found if one exists. If the solution with all facilities open is infeasible, the problem itself is infeasible and the algorithm does not proceed.

The RFLP-MFC does not present the feasibility issues that the RPMP-MFC does:

the feasibility of the problem can be determined by opening all of the facilities and checking whether the resulting solution is feasible. This solution can be used as a starting feasible solution for the tabu search heuristic, or as the upper bound in the subgradient optimization step-size calculation. This is a major advantage of the RFLP-MFC over the RPMP-MFC. Another advantage is that allowing the number of facilities to vary adds an extra degree of freedom to improve the reliability of a solution: open more facilities. This results in more interesting tradeoff curves, since solutions with very different reliability (and cost) can be obtained by varying V^*. (In our experience, the RPMP-MFC is often feasible only for a limited range of V^*, in which only a few solutions are produced.)

5.8 Definitions of Failure Cost

Throughout this chapter, we used the definition of failure cost used in the introduction: the total system-wide transportation cost after a facility fails. However, there are several other ways to model failure costs. For example, failure costs might be defined as the *increase* in cost when a facility fails; the reliability constraints would be formulated as

$$\sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk} \leq V^* \qquad \forall j \in J. \tag{5.94}$$

This definition leads to the following problem. Suppose there are three open facilities; a customer is 1 mile away from facility 1, 100 miles away from facility 2, and 101 miles from facility 3. If constraints (5.94) are used, it may be necessary to assign the customer to facility 2 as a primary facility and facility 3 as a secondary facility so that the increase in cost is small. No firm would assign customers in this way; to prohibit such solutions, a new set of constraints would be required that significantly complicate the formulation and make the relaxations discussed above more difficult to solve.

One could also define failure cost as the *percentage* increase in cost after a facility fails:

$$\sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i (d_{ik} - d_{ij}) Y_{ijk} \leq V^* \left(\sum_{i \in I} \sum_{k \in J} \sum_{l \in J} h_i d_{ik} Y_{ikl} \right) \qquad \forall j \in J. \tag{5.95}$$

This definition is appealing because it allows larger (dollar) increases for higher-volume facilities, but it leads to even greater complications than the previous definition since the right-hand side includes variables, not just a constant.

In the RFLP-MFC, our definition of failure cost includes transportation cost only. One might include fixed costs, as well, to account for the mortgage or lease payments that are still due even while the facility is non-operational:

$$\sum_{j \in J} f_j X_j + \sum_{i \in I} \sum_{\substack{k \in J \\ k \neq j}} \sum_{l \in J} h_i d_{ik} Y_{ikl} + \sum_{i \in I} \sum_{k \in J} h_i d_{ik} Y_{ijk} \leq V^* \qquad \forall j \in J. \tag{5.96}$$

This modification can be incorporated into the LLR and ALR relaxations, but not into the variable-splitting relaxation of the separable formulation, since the reliability constraints are no longer separable once the X_j are added. On the other hand, adding the fixed costs makes the hybrid relaxation applicable to the RFLP-MFC because the left-hand side of the reliability constraints can once again be re-written as the objective function plus a separable term.

In some cases, the transportation cost after a failure may be different from the transportation cost under normal circumstances, for example, because shipments must be arranged with freight companies on an emergency basis. If this is the case, the cost coefficients in the reliability constraints can easily be modified to reflect the alternate cost structure. No change is required in any of the solution methods.

All of these are legitimate definitions of failure cost. The choice of a failure cost definition is a modeling question that must be decided based on the situation at hand.

5.9 Hedge Set Formulation

Suppose that instead of hedging against the failure of individual facilities, we want to hedge against certain sets of facilities failing. For example, maybe the firm is concerned about strikes in Detroit and Dallas, or maybe weather in Anchorage, Fargo, and Bangor often force DC closures and the firm is concerned about the possibility of all three failing at once. Let $\mathcal{S}$ be a collection of subsets of J; we want to hedge against the failure of the sets $S \in \mathcal{S}$. We call the elements S of $\mathcal{S}$ "hedge sets." The facilities in a given hedge set may be ones that are *likely* to fail simultaneously (e.g., because they are served by the same labor union), or simply ones whose simultaneous failure, while unlikely, would be catastrophic (e.g., because they are all major hubs).

For each $S \in \mathcal{S}$, we need to specify a backup site in case customer i's primary facility is in S and all facilities in S become unavailable. Define the following:

$$X_j = \begin{cases} 1, & \text{if we locate a facility at candidate site } j \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, & \text{if facility } j \text{ is selected to be customer } i\text{'s primary facility} \\ 0, & \text{otherwise} \end{cases}$$

$$Z_{iSj} = \begin{cases} 1, & \text{if customer } i\text{'s primary facility is in } S \text{ and facility } j \text{ is selected to be customer} \\ & i\text{'s backup facility if all facilities in } S \text{ fail,} \\ 0, & \text{otherwise} \end{cases}$$

Z_{iSj} is defined for all $i \in I,\ S \in \mathcal{S},\ j \notin S$. Suppose customer i is assigned to primary facility j, and that $S = \{j, k, l\}$ is a hedge set. Customer i does not need a backup if j fails, only if j, k, *and* l fail. The assumption is that any customer served by j can still be adequately served when j fails, provided that k or l is still operational. The hedge-set formulation is as follows:

$$\text{(HEDGE)} \qquad \text{minimize} \qquad \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} \tag{5.97}$$

$$\text{subject to} \qquad \sum_{j \in J} Y_{ij} = 1 \qquad \forall i \in I \tag{5.98}$$

$$\sum_{j \in J \setminus S} Y_{ij} + \sum_{j \in J \setminus S} Z_{iSj} = 1 \qquad \forall i \in I, \forall S \in \mathcal{S} \tag{5.99}$$

$$Y_{ij} \le X_j \qquad \forall i \in I, \forall j \in J \tag{5.100}$$

$$Z_{iSj} \le X_j \qquad \forall i \in I, \forall S \in \mathcal{S}, \forall j \in J \setminus S \tag{5.101}$$

$$\sum_{j \in J} X_j = P \tag{5.102}$$

$$\sum_{i \in I} \sum_{j \in J \setminus S} h_i d_{ij} (Y_{ij} + Z_{iSj}) \le V* \qquad \forall S \in \mathcal{S} \tag{5.103}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{5.104}$$

$$Y_{ij} \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{5.105}$$

$$Z_{iSj} \in \{0,1\} \qquad \forall i \in I, \forall S \in \mathcal{S}, \forall j \in J \setminus S \tag{5.106}$$

The objective function (5.97) sums the demand-weighted distance between each customer and its primary facility. Constraints (5.98) require each customer to be assigned to some primary facility. Constraints (5.99) say that for each customer i and each hedge set S, either i's primary facility is not in S or we must choose some backup facility j not in S to serve i in case all the facilities in S fail. Constraints (5.100) and (5.101) say that a customer cannot be assigned to a facility (primary or backup, respectively) that has not been opened. Constraint (5.102) requires P facilities to be opened. The reliability constraints (5.103) say that if all facilities in a hedge set S fail, the total transportation cost after customers assigned to facilities in S are reassigned must be no greater than V^*. For each hedge set S, the summation on the left-hand side of these constraints sums the cost of serving customers from their primary facilities (if their primary facilities are not in S) or their backup facilities (otherwise). Constraints (5.104)–(5.106) are standard integrality constraints.

If $\mathcal{S} = \{\{j\}|j \in J\}$, this formulation is functionally equivalent to the RPMP-MFC; that is, the two formulations will have the same optimal objective values, and their solutions will be in 1–1 correspondence, but their structures will be different. We have not yet developed solution methods for the hedge set formulation. Solving this model will be a subject for future research.

5.10 Computational Results

5.10.1 Comparison of LP Bounds

We compared the LP relaxations of the three formulations for the RPMP-MFC for 10 randomly generated test problems with 20 nodes and $P = 5$. We also compared the LP relaxations for the RFLP-MFC for 10 randomly generated problems with 20 nodes. We chose to use 20-node problems so that optimal IP solutions could be found using CPLEX in a reasonable amount of time for the sake of comparison. The test problems were generated by drawing integer demands from $U[0, 1000]$, latitudes and longitudes from $U[0, 1]$, and integer fixed costs (for the RFLP-MFC) from $U[300, 1000]$. Transportation costs are set equal to the Euclidean distance between facilities and customers. All nodes serve as both customers and potential facility locations. For each problem, we tested several values of V^*.

The results are reported in Table 5.3. Each row represents the average from the 10 random problems, and each group of rows (separated by lines) uses the same data, but with different values of V^*. The first two columns indicate the problem (RPMP-MFC or RFLP-MFC) and the value of V^*. The column marked "IP Value" lists the average optimal IP objective value among the problems that were feasible; the column marked "IP # Inf" lists the number of problems (out of 10) that were feasible. The remaining sets of columns list, for each of the three formulations (weak, strong, and separable), the

Table 5.3: MFC Models: Comparison of LP bounds.

		IP		Weak			Strong			Separable		
Problem	V^*	Value	# Inf	Value	% Gap	# Inf	Value	% Gap	# Inf	Value	% Gap	# Inf
RPMP	3000	985.6	0	0.0	100.0%	0	984.4	0.1%	0	984.4	0.1%	0
RPMP	2000	960.8	3	0.0	100.0%	0	987.7	4.5%	0	989.5	4.5%	0
RPMP	1600	829.8	5	0.0	100.0%	0	1002.3	2.0%	0	972.4	1.9%	1
RPMP	1200	759.3	8	0.0	100.0%	0	765.5	7.3%	6	781.4	7.0%	6
RPMP	800	—	10	0.0	—	0	676.8	—	9	—	—	10
RFLP	20000	3176.2	0	675.3	78.6%	0	3176.2	0.0%	0	3176.2	0.0%	0
RFLP	10000	3176.2	0	675.3	78.6%	0	3176.2	0.0%	0	3176.2	0.0%	0
RFLP	4000	3180.7	0	675.3	78.6%	0	3176.6	0.1%	0	3176.7	0.1%	0
RFLP	2400	3334.6	0	675.3	79.6%	0	3225.0	3.2%	0	3235.7	2.9%	0
RFLP	2000	3598.0	0	675.3	81.1%	0	3319.8	7.5%	0	3351.7	6.7%	0
RFLP	1600	3910.6	0	675.3	82.5%	0	3552.8	9.0%	0	3610.4	7.5%	0
RFLP	1200	4571.2	0	675.3	85.1%	0	4053.6	11.2%	0	4134.9	9.5%	0
RFLP	800	5756.5	0	675.3	88.2%	0	5145.5	10.6%	0	5232.0	9.1%	0
RFLP	400	8254.9	0	900.4	89.1%	0	7740.5	6.2%	0	7796.5	5.5%	0

average LP relaxation bound, the average percent deviation from the IP value, and the number of problems (out of 10) whose LP relaxation is infeasible. Note that the "Value" and "% Gap" columns are averaged over those problems whose LP relaxation is feasible. It is a desired property of an LP relaxation that it is itself infeasible when the IP is infeasible; this is the motivation for providing the "# Inf" columns. Since the average bounds listed in the "Value" columns include problems whose IP is feasible but whose LP is infeasible, these columns may contain average LP bounds that are greater than the average IP bounds. The "% Gap" columns, on the other hand, do not include infeasible IPs with feasible LPs. All problems (IP and LP) were solved using AMPL/CPLEX 5.0.

The weak formulation is indeed extremely weak, producing bounds of 0 for the RPMP-MFC and gaps of 80%–90% for the RFLP-MFC. The separable formulation generally dominates the strong formulation (both in tightness of the bounds an in ability to detect infeasibility). Both of these formulations, however, provide bounds that are still too weak to make straightforward branch-and-bound a practical approach. It is also worth pointing out that the separable formulation took considerably longer to solve than did the strong formulation; for example, the RPMP-MFC problems took several minutes each on a Sun workstation for the separable formulation and roughly half a minute each for the strong formulation.

5.10.2 Comparison of Relaxation Bounds

We tested the ALR, hybrid, and variable-splitting relaxations on the 20-node data sets described in Section 5.10.1. Our interest was in the tightness of the bounds provided by the relaxations, not in the entire branch-and-bound algorithm, so no branching was performed. In each case, the Lagrangian process terminated when the optimality gap was less than 0.1%, when 1200 iterations had elapsed, or when $\alpha < 10^{-8}$ (see Table 3.1). The results are summarized in Table 5.4. The first two columns indicate the problem (RPMP-MFC or RFLP-MFC) and the value of V^*. The third column lists the average optimal IP value for those problems (among the 10 test problems per row) that are feasible. The remaining columns list, for each relaxation, the average lower bound found (in the "LB" column), the average percentage gap between the lower and upper bounds for those problems for which a feasible solution was found ("Gap"), and the average percentage gap between the lower bound and the optimal IP value for feasible problems ("IP Gap"). (The hybrid relaxation section is empty for the RFLP-MFC since it is not applicable to that problem.)

The hybrid relaxation consistently outperforms the ALR relaxation for the RPMP-MFC, yielding tighter bounds overall. Still, the bounds are not sufficiently tight to make this a practical method for solving problems to optimality. The average IP gap of 2284%

Table 5.4: MFC Models: Comparison of relaxation bounds.

			ALR Relaxation			Hybrid Relaxation			Variable-Splitting		
Problem	V^*	IP	LB	Gap	IP Gap	LB	Gap	IP Gap	LB	Gap	IP Gap
RPMP	3000	985.6	962.3	4.8%	2.7%	983.9	0.2%	0.2%	473.6	7.8%	6.5%
RPMP	2000	960.8	853.9	4.1%	8.9%	977.2	0.3%	5.7%	525.5	10.7%	12.1%
RPMP	1600	829.8	-497.4	7.3%	2284.0%	961.7	5.6%	3.1%	-139.7	20.2%	19.8%
RPMP	1200	759.3	682.6	16.7%	68.3%	1109.0	4.9%	11.6%	98.3	22.1%	26.9%
RPMP	800	—	1963.5	—	—	912.8	—	—	803.9	—	—
RFLP	20000	3176.2	3173.4	0.1%	0.1%				2709.5	21.7%	17.2%
RFLP	10000	3176.2	3173.4	0.1%	0.1%				2698.2	22.0%	17.9%
RFLP	4000	3180.7	3170.3	0.6%	0.3%				1350.0	24.4%	18.6%
RFLP	2400	3334.6	3203.8	7.7%	4.1%				1903.1	139.3%	104.0%
RFLP	2000	3598.0	3285.1	14.5%	9.5%				2309.6	87.4%	70.7%
RFLP	1600	3910.6	3484.6	22.5%	12.1%				2810.0	133.5%	101.3%
RFLP	1200	4571.2	3964.4	21.0%	15.3%				3618.4	93.6%	79.8%
RFLP	800	5756.5	5040.7	20.0%	14.5%				5095.5	15.3%	13.5%
RFLP	400	8254.9	7558.5	10.6%	9.3%				7842.2	5.9%	5.7%

for the ALR relaxation for the RPMP-MFC with $V^* = 1600$ is primarily due to one feasible problem for which no feasible solution was found (so it was not included in the "Gap" column) and for which only a very weak lower bound was found.

The variable-splitting relaxation did not perform well in our computational tests, producing bounds significantly lower than the LP relaxation of the separable formulation, when in theory the Lagrangian relaxation should outperform the LP relaxation. We can identify two possible reasons for the poor performance of this relaxation and the somewhat disappointing performance of the hybrid relaxation. The first is that, as we discussed in Section 4.5.3.2, the objective function coefficients of the MCKPs must be modified to make them non-negative; problems solved to 0.1%-optimality may have significantly larger gaps after the data are converted back into their original form. The second problem may lie in the multiplier-updating routine. Future research will be required to pinpoint the source of the error. In general, Lagrangian relaxation methods depend heavily on good initial multipliers and good multiplier-updating procedures; our relative inexperience with hybrid relaxation and variable-splitting (as opposed to more

Table 5.5: MFC Models: Comparison of relaxation times.

Problem	V^*	ALR Relaxation # Iter	Time	Time/Iter	Hybrid Relaxation # Iter	Time	Time/Iter	Variable-Splitting # Iter	Time	Time/Iter
RPMP	3000	1081.0	33.1	0.031	173.5	8.1	0.051	1136.8	173.5	0.149
RPMP	2000	1080.9	31.9	0.030	443.5	20.1	0.044	1079.1	149.3	0.136
RPMP	1600	1058.1	32.6	0.032	592.1	26.9	0.072	955.4	133.2	0.139
RPMP	1200	1008.9	32.7	0.033	485.6	21.6	0.066	643.9	109.0	0.219
RPMP	800	483.7	15.6	0.038	79.4	3.2	0.084	142.5	24.6	0.230
RFLP	20000	51.4	1.1	0.020				841.2	43.0	0.050
RFLP	10000	51.4	1.2	0.024				833.8	41.8	0.050
RFLP	4000	897.1	21.0	0.022				1009.8	174.4	0.154
RFLP	2400	1121.2	18.8	0.017				942.0	226.5	0.203
RFLP	2000	1200.0	20.8	0.017				986.8	231.2	0.207
RFLP	1600	1200.0	22.9	0.019				939.8	235.9	0.221
RFLP	1200	1200.0	27.9	0.023				863.0	207.7	0.233
RFLP	800	1200.0	25.4	0.021				770.3	170.2	0.230
RFLP	400	1200.0	27.0	0.023				729.0	232.8	0.314

straightforward Lagrangian relaxation) may be contributing to our failure to attain good bounds from these methods.

The hybrid relaxation provides tighter bounds than the ALR relaxation, but this increase in accuracy comes with an increase in computation time since this relaxation involves solving MCKPs. Table 5.5 lists the average number of iterations spent solving the problems, the average CPU time (in seconds, on a Dell Inspiron 7500 notebook computer with a 500 MHz Pentium III processor and 128 MB memory), and the average time per iteration. Although the hybrid relaxation requires more time per iteration, it ultimately requires less time than the ALR relaxation since it reaches optimality (or proves infeasibility) in fewer iterations.

5.10.3 Tabu Search Heuristic Performance

We tested the tabu search heuristic on the same data sets described in Section 5.10.1 (10 for the RPMP-MFC, 10 for the RFLP-MFC). The tabu search heuristic was run once for each instance. Table 5.6 summarizes the performance of the heuristic and compares it

Table 5.6: Tabu search heuristic performance.

		Tabu Search						CPLEX		
Problem	V^*	Value	Avg % Gap	SD % Gap	# Inf	Time	# Iter	Value	# Inf	Time
RPMP	3000	997.1	1.0%	1.5%	0	0.05	73.3	985.6	0	16.4
RPMP	2000	814.0	0.1%	0.2%	5	0.05	88.8	960.8	3	105.0
RPMP	1600	838.5	1.1%	1.9%	5	0.04	92.9	829.8	5	43.4
RPMP	1200	759.3	0.0%	0.0%	8	0.05	96.2	759.3	8	39.6
RPMP	800	—	—	—	10	0.05	100.0	—	10	40.3
RFLP	20000	3181.9	0.2%	0.4%	0	0.03	55.0	3176.2	0	11.7
RFLP	10000	3181.4	0.2%	0.4%	0	0.03	55.6	3176.2	0	11.8
RFLP	4000	3204.5	0.7%	1.7%	0	0.03	59.1	3180.7	0	12.9
RFLP	2400	3474.1	4.0%	5.6%	0	0.02	53.3	3334.6	0	23.7
RFLP	2000	3680.7	2.3%	3.8%	0	0.05	64.6	3598.0	0	39.2
RFLP	1600	4106.1	4.9%	4.4%	0	0.04	62.0	3910.6	0	40.2
RFLP	1200	4958.2	8.3%	6.6%	0	0.03	74.1	4571.2	0	43.1
RFLP	800	6357.0	9.9%	9.5%	0	0.03	63.9	5756.5	0	35.3
RFLP	400	9082.3	9.9%	9.1%	0	0.03	62.0	8254.9	0	13.9

to using CPLEX to solve the strong formulation. The first two columns are as in Table 5.3. The next set of columns describe the performance of the tabu search heuristic: "Value" indicates the average objective value of the feasible solutions found; "Avg % Gap" and "SD % Gap" give the mean and standard deviation of the percentage gap from the optimal solution found using CPLEX, among the problems for which feasible solutions were found by the heuristic; "# Inf" indicates the number of problems (out of 10) for which no feasible solution was found by the heuristic; "Time" gives the average CPU time (in seconds) spent on each problem; and "# Iter" lists the average number of tabu search iterations executed before the heuristic terminated. The final three columns list the average optimal IP objective value (among the feasible problems), the number of problems (out of 10) that are infeasible, and the average time required by CPLEX to solve each problem (excluding I/O time). Note that the tabu search algorithm was executed on a notebook computer, while CPLEX was run on a Sun workstation.

The tabu search heuristic generally finds solutions within a few percent of optimal. It also executes very quickly, requiring well under 0.1 seconds per problem. Allowing the

Table 5.7: Tabu search heuristic performance: 100-node RFLP-MFC problem.

V^*	Value	Time	# Iter
100000	9306.5	5.0	111.7
50000	9298.2	5.9	132.3
20000	9247.6	5.4	120.4
12000	9254.5	5.3	120.6
10000	9290.2	6.4	143.6
8000	9753.2	5.8	127.8
6000	9693.3	6.4	139.9
4000	12105.7	6.0	131.4
2000	22824.9	6.1	136.7

heuristic to run for more iterations may produce lower-cost solutions while still maintaining a reasonably tradeoff between CPU time and solution quality; however, since the heuristic generally terminated before the maximum of 100 iterations was reached, more emphasis should be placed on diversification if such a strategy is attempted.

We also tested the tabu search heuristic on 10 problems of 100 nodes each, generated randomly as described in Section 5.10.1. Only the RFLP-MFC was solved. The purpose of these tests is to demonstrate the execution time of the heuristic on larger problems; since these problems are far too large to solve optimally using CPLEX, we cannot compute the quality of the solutions found. The results of these tests are given in Table 5.7. Note that even for these larger problems, the heuristic executes in under 10 seconds.

5.10.4 Tradeoff Curves

We generated a tradeoff curve for the RFLP-MFC using a randomly generated, 100-node problem, following the method described in Section 5.6. The problems were solved using the tabu search heuristic. The curve is pictured in Figure 5.3. The optimal UFLP solution ($V^* = \infty$) is the left-most point on the curve. The left portion of the tradeoff curve is steep, indicating that large improvements in reliability may be attained with small increases in UFLP cost. The flat right-most portion is of less interest, since it contains very expensive (though very reliable) solutions. We find this shape to be typical of the tradeoff curves produced by the RFLP-MFC. The RPMP-MFC produces much less interesting tradeoff curves since the fixed number of facilities offers much less flexibility to improve reliability.

The 10 least expensive solutions are listed in Table 5.8, along with the number of facilities open in each and the solutions' relationships to the optimal UFLP solution. The table indicates that, for example, a 15% reduction in maximum failure cost is possible with only a 3% increase in UFLP cost. This solution requires one additional facility, but this comes at minimal extra cost. Neither of the two facilities open in solution 1 is open in solution 2. The increases in failure cost between solutions 6 and 7 and between solutions 8 and 9 are simply due to the fact that the problems were solved heuristically, rather than exactly; this also explains the kinkiness of the tradeoff curve itself. If the problems had been solved optimally, the curve would be even steeper at the left and flatter at the right. The tradeoff curve took 996 seconds to generate in its entirety on a notebook computer.

5.11 Chapter Summary

In this chapter we presented two new models that incorporate reliability into classical facility location problems. These models arose from a realization that supply chains are

Figure 5.3: RFLP-MFC tradeoff curve for 100-node data set.

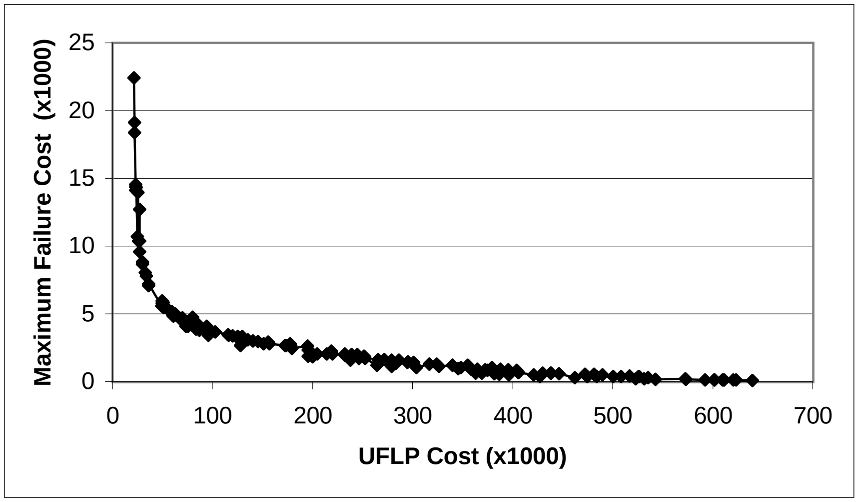


Table 5.8: First 10 solutions in curve: RFLP-MFC.

Soln #	Obj 1	Obj 2	% Increase Obj 1	% Decrease Obj 2	# Locations
1	21181	22420	0.0%	0.0%	2
2	21738	19141	2.6%	14.6%	3
3	21879	18374	3.3%	18.0%	3
4	23124	14539	9.2%	35.2%	4
5	23229	14376	9.7%	35.9%	4
6	23343	14122	10.2%	37.0%	4
7	23489	14351	10.9%	36.0%	4
8	24991	10711	18.0%	52.2%	4
9	25095	13974	18.5%	37.7%	5
10	25763	10395	21.6%	53.6%	5

vulnerable to disruptions of all sorts, and that facility location decisions can be critical in reducing the impact of these disruptions. The models both consider the maximum failure cost given a single facility failure. We formulated models based on the P-median problem and the uncapacitated fixed-charge location problem, called the RPMP-MFC and RFLP-MFC, respectively. Key to our formulations is the concept of "backup" assignments, which represent the facilities to which customers are assigned when their primary facilities have failed. In both models, a set of constraints restricts the transportation cost after a failure to be less than a given upper limit. The tradeoff of interest is between the operating cost (the traditional PMP or UFLP objective function) and the maximum failure cost. A tradeoff curve between these two objectives can be constructed using the constraint method of multi-objective programming.

The addition of the reliability constraints to classical facility location models complicates those problems significantly. We proposed several formulations and relaxations of the MFC models, but none has consistently provided tight enough bounds to enable its use in an exact algorithm. As a result, we also proposed a tabu search heuristic that provides good solutions quickly, though with no guarantee of optimality. Clearly, finding an exact algorithm to solve these problems is of great importance and will be the subject of future research.

The main disadvantage of the MFC models (other than the computational aspects) is that they assume that only a single facility can fail at a time. This may be reasonable for certain applications—for example, if failures are extremely unlikely or can be quickly recovered from—but not for others. One way around this problem is to hedge against failures not of individual facilities but of all facilities in some pre-specified sets. This is the basis for the "hedge set" formulation discussed in Section 5.9. This formulation, however, requires the hedge sets of interest to be identified exogenously. Moreover, neither the individual-failure and the hedge set models take into account the probability or the duration of failures. The expected failure cost models discussed in the next chapter address these issues in that they allow the simultaneous failures of multiple facilities, based on probabilistic information.

Chapter 6

Reliability Models for Facility Location: Expected Failure Cost

6.1 Introduction

In Chapter 5, we introduced reliability models that hedge against the worst-case failure among a set of facilities. In this chapter we consider the *expected* failure cost, given a certain probability that each facility fails. Unlike the maximum failure cost case, in this chapter we assume that multiple failures may occur simultaneously. The goal is to choose facility locations so that the system is inexpensive to construct and operate day-to-day, but also so that the long-run expected cost due to failures is minimized. Certain facilities may be designated as "non-failable." In our work with a major manufacturer of durable goods, the facilities that may fail represent warehouses owned by independent distributors who occasionally "defect" from the company or go out of business. The non-failable warehouses are those owned by the company; these are assumed to remain loyal to the firm and will not fail. In other applications, the non-failable facilities may represent those located in favorable weather areas, those served by unions with which the firm has a particularly strong relationship, or other facilities deemed to have a negligible probability of failure.

Whether to use the maximum failure cost (MFC) or expected failure cost (EFC) models in a given situation is a question that must be answered by the modeler. Since the MFC models hedge against the worst case, they may be appealing to decision makers who are evaluated based on the system's performance in a short time period. On the other hand, the MFC models ignore the likelihood or duration of a failure at a given facility, leading to solutions that plan against an extreme and possibly unlikely event. The EFC models are more realistic in the sense that they incorporate probabilistic information into the objective function, optimizing based on the modeler's preference between planning against the day-to-day and planning against failures. In addition, we have developed exact solution methods for the EFC problems that perform much better than those for the MFC, so the EFC may be a more practical modeling choice for performance reasons.

The MFC problems are multi-objective models: one objective (operating cost) is explicitly optimized in the objective function while the other (failure cost) is implicitly optimized via a constraint. Tradeoff curves are generated using the constraint method by systematically tightening the failure cost constraint. The EFC problems are also multi-objective models, but in this case both objectives are explicitly optimized in the objective function, weighted by coefficients that the modeler can set to express her preference between the two objectives. Tradeoff curves for the EFC problems are generated using the weighting method, systematically varying the weighting coefficient.

As in Chapter 5, we will first discuss the expected failure cost version of the reliability P-median problem (the RPMP-EFC), then indicate how similar ideas can be applied to formulate and solve the expected failure cost version of the reliability fixed-charge location problem (the RFLP-EFC).

6.2 Formulation

6.2.1 Notation

As in the RPMP-MFC, we let I represent the set of customers and J the set of potential facility sites. Let NF be the set of facilities that may not fail (we refer to these as "non-failable" facilities) and let F be the set of facilities that may fail ("failable" facilities). Note that $NF \cup F = J$ and $NF \cap F = \emptyset$.

The notation for the RPMP-EFC is as follows:

Parameters

h_i = demand per period for customer $i \in I$

d_{ij} = per-unit cost to ship from facility $j \in J$ to customer $i \in I$

P = number of facilities to open ($P \geq 2$)

α = weight on objective 1 ($0 \leq \alpha \leq 1$)

q = probability that a failable facility will fail ($0 \leq q \leq 1$)

θ_i = cost of not serving customer $i \in I$, per unit of demand

The parameter V^* used in the MFC problems is not needed since the failure costs are no longer constrained; in its place is an objective function parameter α ($0 \leq \alpha \leq 1$) that can be varied to generate a tradeoff curve; see Section 6.4.

Each facility in F has the same probability q of failing, which is interpreted as the long-run fraction of time the facility is non-operational. In some cases, q may be estimated based on historical data (e.g., for weather-induced failures), while in others q must be estimated subjectively (e.g., for failures due to the defection of third-party distributors). Our model is most easily interpreted as an infinite-horizon model in which q represents the fraction of time that a facility has failed. However, if the modeler has in mind a particular time horizon T, then q may be used to capture probabilistic information about the timing of the failures. For example, suppose each facility has a 0.1 probability of failing, and if it fails, it will fail in period 1 with probability 0.3 and in period 2 with probability 0.7. Then the expected fraction of time the facility will be non-operational is given by $(0.1 \times 0.3 \times T + 0.1 \times 0.7 \times (T-1))/T$.

Associated with each customer i is a cost θ_i that represents the cost of not serving the customer, per unit of demand. θ_i may be a lost-sales cost, or the cost of serving i by purchasing product from a competitor on an emergency basis. This cost is incurred if all open facilities have failed (and thus no facilities are available to serve customer i), or if θ_i is less than the cost of assigning i to any of the existing facilities. To model this, we add an "emergency" facility u that is non-failable ($u \in NF$) and has transportation cost $d_{iu} = \theta_i$ to customer $i \in I$. We force $X_u = 1$ and replace P with $P + 1$. From this point forward, we assume that the emergency facility has been handled in this way, though for simplicity we continue to formulate the problem as a P-median, rather than as a $(P+1)$-median, problem.

The strategy behind the formulation of the RPMP-EFC is to assign each customer to a primary facility that will serve it under normal circumstances, as well as to a *set* of backup facilities that serve it when the primary facility has failed. Since multiple failures may occur simultaneously, each customer needs a first backup facility in case its primary facility fails, a second backup in case its first backup fails, and so on. (This is in contrast to the RPMP-MFC, in which only a single backup is needed since only one facility may fail at a time.) However, if a customer is assigned to a non-failable facility as its nth backup, it does not need any further backups.

There are two sets of decision variables in the model, location variables (X) and assignment variables (Y):

$$X_j = \begin{cases} 1, & \text{if a facility is opened at location } j \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ijr} = \begin{cases} 1, & \text{if demand node } i \text{ is assigned to facility } j \text{ as a level-}r \text{ assignment} \\ 0, & \text{otherwise} \end{cases}$$

A "level-r" assignment is one for which there are r closer failable facilities that are open. If $r = 0$, this is a primary assignment; otherwise, it is a backup assignment. Each customer i has a level-r assignment for each $r = 0, \ldots, P-1$, unless i is assigned to a level-s facility that is non-failable, where $s < r$. In other words, customer i is assigned to one facility at level 0, another facility at level 1, and so on until i has been assigned to a non-failable facility at some level (there must be such a facility since $u \in NF$ is always open).

6.2.2 Objectives

We formulate this problem as a multi-objective problem. The objectives are as follows:

$$w_1 = \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij0}$$
$$w_2 = \sum_{i \in I} h_i \left[\sum_{j \in NF} \sum_{r=0}^{P-1} d_{ij} q^r Y_{ijr} + \sum_{j \in F} \sum_{r=0}^{P-1} d_{ij} q^r (1-q) Y_{ijr} \right].$$

Objective w_1 computes the operating cost—the P-median cost of serving customers from their primary facilities. Objective w_2 computes the expected failure cost: each customer i is served by its level-r facility (call it j) if the r closer facilities have failed (this occurs with probability q^r) and if j itself has not failed (this occurs with probability $1-q$ if $j \in F$ and with probability 1 if $j \in NF$). Note that by the definition of level-r, all r closer facilities are failable.

Although we refer to w_2 as the "expected failure cost," we are careful to point out that w_2 also includes the transportation cost when no facilities have failed (i.e., the level-0 assignments). Certainly, there are ways to define reliability other than that given in w_2. For example, if the desired tradeoff is between PMP cost and expected transportation cost only after a failure, then the "primary" transportation cost can be omitted from w_2 by starting the summation indices at $r = 1$ rather than $r = 0$. It is also possible that the transportation costs for backup assignments are different from those for primary assignments as suggested in Section 5.8; in this case, the coefficients for Y_{ijr} would be changed from d_{ij} to some other cost for $r > 0$. Either of these modifications can be handled easily using the solution method described below.

Our model minimizes a weighted sum $\alpha w_1 + (1-\alpha) w_2$ of the two objectives, where $0 \leq \alpha \leq 1$. By solving the problem for various values of α, one can generate a tradeoff curve between the operating cost and the expected failure cost using the weighting method of multi-objective programming (see Section 6.4).

6.2.3 Integer Programming Formulation

The reliability P-median problem is formulated as follows:

$$\text{(RPMP-EFC)} \quad \text{minimize} \quad \alpha w_1 + (1-\alpha) w_2 \tag{6.1}$$

$$\text{subject to} \quad \sum_{j \in J} Y_{ijr} + \sum_{j \in NF} \sum_{s=0}^{r-1} Y_{ijs} = 1 \qquad \forall i \in I, r = 0, \ldots, P-1 \tag{6.2}$$

$$Y_{ijr} \leq X_j \qquad \forall i \in I, j \in J, r = 0, \ldots, P-1 \tag{6.3}$$

$$\sum_{j \in J} X_j = P \tag{6.4}$$

$$\sum_{r=0}^{P-1} Y_{ijr} \leq 1 \qquad \forall i \in I, j \in J \tag{6.5}$$

$$X_u = 1 \tag{6.6}$$

$$X_j \in \{0,1\} \quad \forall j \in J \tag{6.7}$$

$$Y_{ijr} \in \{0,1\} \quad \forall i \in I, j \in J, r = 0, \ldots, P-1 \tag{6.8}$$

The objective function (6.1) is straightforward. Constraints (6.2) require that for each customer i and each level r, either i is assigned to a level-r facility or it is assigned to a level-s facility ($s < r$) that is non-failable. (By convention we take $\sum_{s=0}^{r-1} Y_{ijs} = 0$ if $r = 0$.) Constraints (6.3) prohibit an assignment to a facility that has not been opened. Constraint (6.4) requires P facilities to be opened. Constraints (6.5) prohibit a customer from being assigned to a given facility at more than one level. Constraint (6.6) requires the emergency facility u to be opened. Constraints (6.7) and (6.8) are standard integrality constraints.

If $\alpha = 1$, solving (RPMP-EFC) is equivalent to solving the PMP (the backup assignments are irrelevant), so the RPMP-EFC is NP-hard (as is the RFLP-EFC, discussed later).

For notational convenience, we can write the objective function as

$$\sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1} \psi_{ijr} Y_{ijr}, \tag{6.9}$$

where

$$\psi_{ijr} = \begin{cases} \alpha h_i d_{ij} + (1-\alpha) h_i d_{ij} = h_i d_{ij}, & \text{if } r = 0 \text{ and } j \in NF \\ \alpha h_i d_{ij} + (1-\alpha) h_i d_{ij} (1-q), & \text{if } r = 0 \text{ and } j \in F \\ (1-\alpha) h_i d_{ij} q^r, & \text{if } r > 0 \text{ and } j \in NF \\ (1-\alpha) h_i d_{ij} q^r (1-q), & \text{if } r > 0 \text{ and } j \in F \end{cases}$$

One might suspect that for large α, the weight on the backup assignments may be larger than that on the primary assignments, in which case it may be optimal to assign customers to primary facilities that are farther than their backup facilities, a situation we would want to prohibit. The next theorem, however, demonstrates that such a situation cannot occur.

Theorem 6.1 *In any optimal solution to (RPMP-EFC), if* $Y_{ijr} = Y_{i,k,r+1} = 1$ *for* $i \in I$, $j, k \in J$, $0 \le r < P - 1$, *then* $d_{ij} \le d_{ik}$.

Proof. Suppose, for a contradiction, that (X, Y) is an optimal solution to (RPMP-EFC) in which $Y_{ijr} = Y_{i,k,r+1} = 1$ but $d_{ij} > d_{ik}$. We will show that by "swapping" j and k, the objective function will decrease. Since i has a level-$(r + 1)$ facility (k), its level-r facility (j) must be failable.

Suppose first that $k \in F$. These two assignments contribute $\psi_{ijr} + \psi_{i,k,r+1}$ to the objective function. If we assigned i to j at level $r + 1$ and to k at level r, the objective function would change by $(\psi_{ikr} + \psi_{i,j,r+1}) - (\psi_{ijr} + \psi_{i,k,r+1})$. If $r = 0$, then

$$\begin{aligned}(\psi_{ikr} + \psi_{i,j,r+1}) - (\psi_{ijr} + \psi_{i,k,r+1}) =& \alpha h_i d_{ik} + (1-\alpha) h_i d_{ik} (1-q) + (1-\alpha) h_i d_{ij} q (1-q) \\ & - \alpha h_i d_{ij} - (1-\alpha) h_i d_{ij} (1-q) - (1-\alpha) h_i d_{ik} q (1-q) \\ =& \alpha h_i (d_{ik} - d_{ij}) + (1-\alpha) h_i (d_{ik} - d_{ij}) (1-q)^2 \\ <& 0\end{aligned}$$

since $d_{ij} > d_{ik}$ and $0 \le \alpha \le 1$. On the other hand, if $r > 0$, then

$$\begin{aligned}(\psi_{ikr} + \psi_{i,j,r+1}) - (\psi_{ijr} + \psi_{i,k,r+1}) =& (1-\alpha) h_i d_{ik} q^r (1-q) + (1-\alpha) h_i d_{ij} q^{r+1} (1-q) \\ & - (1-\alpha) h_i d_{ij} q^r (1-q) - (1-\alpha) h_i d_{ik} q^{r+1} (1-q) \\ =& (1-\alpha) h_i (d_{ik} - d_{ij}) q^r (1-q)^2 \\ <& 0.\end{aligned}$$

Either way, the objective function is smaller for the revised solution. The case in which

$k \in NF$ is similar, except that in this case, $Y_{i,j,r+1} = 0$ since i's level-r facility is non-failable, resulting in an even larger decrease in cost. This contradicts the assumption that (X, Y) is optimal. □

We note briefly that if the level-0 assignments are excluded from w_2 as discussed on page 212, then Theorem 6.1 only holds when $\alpha \geq \frac{1}{2}$, which is generally the range of interest to decision makers. In this case, the algorithm given below may still be valid for particular instances, even if $\alpha < \frac{1}{2}$. If the algorithm returns a solution for which the distance ordering is obeyed, it is optimal; but the algorithm cannot enforce the distance ordering if it is not naturally optimal to do so.

6.3 Lagrangian Relaxation

6.3.1 Lower Bound

We solve (RPMP-EFC) by relaxing constraints (6.2) using Lagrangian relaxation. For given Lagrange multipliers λ, the subproblem is as follows:

(RPMP-EFC-LR$_\lambda$)

$$\text{minimize} \quad z(\lambda) = \sum_{i \in I} \sum_{j \in J} \sum_{r=0}^{P-1} \psi_{ijr} Y_{ijr} + \sum_{i \in I} \sum_{r=0}^{P-1} \lambda_{ir} \left(1 - \sum_{j \in J} Y_{ijr} - \sum_{j \in NF} \sum_{s=0}^{r-1} Y_{ijs} \right) \quad (6.10)$$

$$\text{subject to} \quad Y_{ijr} \leq X_j \quad \forall i \in I, j \in J, r = 0, \ldots, P-1 \quad (6.11)$$

$$\sum_{j \in J} X_j = P \quad (6.12)$$

$$\sum_{r=0}^{P-1} Y_{ijr} \leq 1 \quad \forall i \in I, j \in J \quad (6.13)$$

$$X_u = 1 \tag{6.14}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{6.15}$$

$$Y_{ijr} \in \{0,1\} \qquad \forall i \in I, j \in J, r = 0, \ldots, P-1 \tag{6.16}$$

The objective function (6.10) can be re-written as follows:

$$\sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\psi_{ijr}Y_{ijr} + \sum_{i\in I}\sum_{r=0}^{P-1}\lambda_{ir} - \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\lambda_{ir}Y_{ijr} - \sum_{i\in I}\sum_{r=0}^{P-1}\sum_{j\in NF}\sum_{s=0}^{r-1}\lambda_{ir}Y_{ijs}$$
$$= \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\psi_{ijr}Y_{ijr} + \sum_{i\in I}\sum_{r=0}^{P-1}\lambda_{ir} - \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\lambda_{ir}Y_{ijr} - \sum_{i\in I}\sum_{j\in NF}\sum_{s=0}^{P-1}\sum_{r=0}^{s-1}\lambda_{is}Y_{ijr}$$

(by swapping the indices r and s in the last term)

$$= \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\psi_{ijr}Y_{ijr} + \sum_{i\in I}\sum_{r=0}^{P-1}\lambda_{ir} - \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\lambda_{ir}Y_{ijr} - \sum_{i\in I}\sum_{j\in NF}\sum_{\substack{r=0,\ldots,P-1\\ s=0,\ldots,P-1\\ r<s}}\lambda_{is}Y_{ijr}$$
$$= \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\psi_{ijr}Y_{ijr} + \sum_{i\in I}\sum_{r=0}^{P-1}\lambda_{ir} - \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\lambda_{ir}Y_{ijr} - \sum_{i\in I}\sum_{j\in NF}\sum_{r=0}^{P-1}\left(\sum_{s=r+1}^{P-1}\lambda_{is}\right)Y_{ijr}$$

Therefore, the objective function can be written as

$$\sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{P-1}\tilde{\psi}_{ijr}Y_{ijr} + \sum_{i\in I}\sum_{r=0}^{P-1}\lambda_{ir}, \tag{6.17}$$

where

$$\tilde{\psi}_{ijr} = \begin{cases} \psi_{ijr} - \lambda_{ir}, & \text{if } j \in F \\ \psi_{ijr} - \lambda_{ir} - \left(\sum_{s=r+1}^{P-1}\lambda_{is}\right) = \psi_{ijr} - \sum_{s=r}^{P-1}\lambda_{is}, & \text{if } j \in NF \end{cases} \tag{6.18}$$

For given λ, problem (RPMP-EFC-LR$_\lambda$) can be solved easily. Since the assignment constraints (6.2) have been relaxed, customer i may be assigned to zero, one, or more than one open facility at each level, but it may be assigned to a given facility at at most one level r. Suppose that facility j is opened. Customer i will be assigned to facility j at level r if $\tilde{\psi}_{ijr} < 0$ and $\tilde{\psi}_{ijr} \leq \tilde{\psi}_{ijs}$ for all $s = 0, \ldots, P-1$. Therefore, the benefit of opening facility j is given by

$$\gamma_j = \sum_{i \in I} \min \left\{ 0, \min_{r=0,\ldots,P-1} \{\tilde{\psi}_{ijr}\} \right\}. \tag{6.19}$$

Once the benefits γ_j have been computed for all j, we set $X_j = 1$ for the emergency facility u and for the $P-1$ remaining facilities with the smallest γ_j; we set $Y_{ijr} = 1$ if (1) facility j is open, (2) $\tilde{\psi}_{ijr} < 0$, and (3) r minimizes $\tilde{\psi}_{ijs}$ for $s = 0, \ldots, P-1$. The optimal objective value for (RPMP-EFC-LR$_\lambda$) is $z(\lambda) = \sum_{j \in J} \gamma_j X_j$, and this provides a lower bound on the optimal objective value of (RPMP-EFC).

The benefit γ_j can be computed for a single j in $O(nP)$ time, where $n = |I|$, so all of the benefits can be computed in $O(mnP)$ time, where $m = |J|$. Determining X_j requires sorting the facilities, which takes $O(m \log m)$ time, and determining Y_{ijr} requires $O(nP)$ time, assuming that assignments are stored as a single index j for each i, r rather than as a list of m 0/1 variables. Therefore, the Lagrangian subproblem can be solved for a given λ in $O(mnP + m \log m + nP) = O(mnP)$ time.

6.3.2 Upper Bound

If the solution to (RPMP-EFC-LR$_\lambda$) is feasible for (RPMP-EFC), then it provides both a lower bound and an upper bound, and is in fact optimal for (RPMP-EFC). Otherwise, we construct a feasible solution as follows. First, we open the facilities that are open in the solution to (RPMP-EFC-LR$_\lambda$). Next, we assign customers to the open facilities level by level in increasing order of distance, until a non-failable facility is assigned. (By Theorem 6.1, this is an optimal strategy for assigning customers to a given set of facilities, though the facilities themselves may not be optimal.) If the resulting solution has objective value 1.2UB or less, where UB is the objective value of the best known solution, it becomes a candidate for improvement. One out of every five candidate solutions are passed to a DC exchange heuristic that attempts to improve the solution by opening a facility that is currently closed and closing one that is currently open, similar to the vertex substitution heuristic of Teitz and Bart (1968). The parameters 1.2 and 5 given in the preceding sentences may easily be changed. By increasing the threshold value and/or the frequency with which the DC exchange heuristic executes, one obtains higher-quality solutions but longer run times. Anecdotally, we can report that the heuristic as described here has performed well in our computational tests, finding the optimal solution very quickly (generally within the first 100 Lagrangian iterations), though we have not explicitly recorded the iteration at which the optimal solution is found.

6.3.3 Multiplier Updating

Each value of λ provides a lower bound $z(\lambda)$ on the optimal objective value of (RPMP-EFC). To find the best possible lower bound, we use subgradient optimization, applied in a straightforward manner as described by Fisher (1981, 1985) and Daskin (1995). In particular, at each iteration n we compute a step-size t^n as

$$t^n = \frac{\beta^n(\text{UB} - \mathcal{L}^n)}{\sum_{i\in I}\sum_{r=0}^{P-1}\left(1 - \sum_{j\in J} Y_{ijr} + \sum_{j\in NF}\sum_{s=0}^{r-1} Y_{ijs}\right)^2}, \tag{6.20}$$

where β^n is a constant initialized to 2 and halved when 30 consecutive iterations fail to improve the lower bound, $\mathcal{L}^n$ is the value of $z(\lambda)$ found at iteration n, and UB is the best known upper bound. The multipliers are updated by setting

$$\lambda_{ir}^{n+1} \leftarrow \lambda_{ir}^{n} + t^n\left(1 - \sum_{j\in J} Y_{ijr} + \sum_{j\in NF}\sum_{s=0}^{r-1} Y_{ijs}\right). \tag{6.21}$$

The Lagrangian process terminates when any of the following criteria is met:

- $(\text{UB} - \mathcal{L}^n)/\mathcal{L}^n < \epsilon$, for some optimality tolerance ϵ specified by the user
- $n > n_{\max}$, for some iteration limit $n_{\max}$
- $\beta^n < \beta_{\min}$, for some β limit $\beta_{\min}$

6.3.4 Branch and Bound

If the Lagrangian process terminates with the lower and upper bounds equal (to within ϵ), an ϵ-optimal solution has been found and the algorithm terminates. Otherwise, we use branch-and-bound to close the optimality gap. We branch on the X_j (location) variables. At each branch-and-bound node, the facility selected for branching is the unfixed open facility with the greatest assigned demand. X_j is first forced to 0 and then to 1. Branching is done in a depth-first manner. The tree is fathomed at a given node if the lower bound at that node is within ϵ of the objective function value of the best feasible solution found anywhere in the tree, if P facilities have been forced open, or if $|J| - P$ facilities have been forced closed. The final Lagrange multipliers at a given node are passed to its child nodes and are used as initial multipliers at those nodes.

6.3.5 Variable Fixing

If the Lagrangian procedure terminates at the root node without a proof of optimality, a variable-fixing method similar to that for the SLMRP (see Section 3.2.4) can be used for the RPMP-EFC. Assume for notational convenience that the facilities in $J \setminus \{u\}$ are sorted in increasing order of benefit so that $\gamma_j \leq \gamma_{j+1}$, under a particular set of Lagrange multipliers λ. Let LB be the lower bound (the objective value of (RPMP-EFC-LR$_\lambda$)) under the same λ, and let UB be the best upper bound found. Suppose further that $X_j = 0$ in the solution to (RPMP-EFC-LR$_\lambda$). If

$$\text{LB} + \gamma_j - \gamma_{P-1} > \text{UB} \tag{6.22}$$

then candidate site j *cannot* be part of the optimal solution, so we can fix $X_j = 0$. This is true because if j were forced into the solution, another facility would be forced out; this facility would be the open facility (other than u) with the largest benefit, i.e., facility $P-1$. Clearly $\text{LB} + \gamma_j - \gamma_{P-1}$ is a valid lower bound for the "$X_j = 1$" node (it would be the first lower bound found if we use λ as the initial multipliers at the new child node), so we would fathom the tree at this new node and never again consider setting $X_j = 1$.

Similarly, suppose $X_j = 1$ in the solution to (RPMP-EFC-LR$_\lambda$). If

$$\text{LB} - \gamma_j + \gamma_P > \text{UB} \tag{6.23}$$

then candidate site j *must* be part of the optimal solution since swapping j out and the best closed facility in will result in a solution whose lower bound exceeds the upper bound; therefore, we can fix $X_j = 1$.

We perform these variable-fixing checks twice after processing has terminated at the root node, once using the optimal multipliers λ and once using the most recent multipliers. This procedure is quite effective in forcing variables open or closed because the Lagrangian procedure tends to produce tight lower bounds, making (6.22) or (6.23) hold for many facilities j. The time required to perform these checks is negligible.

6.4 Tradeoff Curves

By systematically varying the objective function weight α and re-solving (RPMP-EFC) for each value, one can generate a tradeoff curve between the two objectives using the weighting method of multi-objective programming (Cohon 1978). The method is as follows:

0. Solve (RPMP-EFC) for $\alpha = 1$ (the pure PMP problem) and for $\alpha = 0$. Add both points to the tradeoff curve.

1. Identify an adjacent pair of solutions on the tradeoff curve that has not yet been considered. Let the objective values of these two solutions be (w_1^1, w_2^1) and (w_1^2, w_2^2). Set $\alpha \leftarrow -(w_2^1 - w_2^2)/(w_1^1 - w_1^2 - w_2^1 + w_2^2)$.

2. Solve (RPMP-EFC) for the current value of α. If the resulting solution is not already on the tradeoff curve, add it.

3. If all adjacent pairs of solutions on the tradeoff curve have been explored, stop. Else, go to 1.

Sample tradeoff curves are shown in Section 5.10.4.

6.5 UFLP-Based Problems

The RPMP-EFC can improve reliability only by choosing a different set of P facilities, not by opening additional ones. In this section, we formulate the expected failure cost version of the Reliability Fixed-Charge Location Problem (RFLP-EFC), which is based on the UFLP. Since the UFLP does not contain a limit on the number of facilities that can be built, the RFLP-EFC adds a degree of freedom for improving reliability, namely, constructing additional facilities.

6.5.1 Formulation

The RFLP-EFC is formulated in a manner similar to the RPMP-EFC. We need one additional parameter: f_j is the fixed cost to construct a facility at location $j \in J$, amortized to the time units used to express demands. Since the number of facilities is not known *a priori* as it is in the RPMP-EFC, we must create assignment variables for levels $r = 0, ..., m-1$, where $m = |J|$. The objectives are given by

$$w_1 = \sum_{j \in J} f_j X_j + \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij0}$$

$$w_2 = \sum_{i \in I} h_i \left[\sum_{j \in NF} \sum_{r=0}^{m-1} d_{ij} q^r Y_{ijr} + \sum_{j \in F} \sum_{r=0}^{m-1} d_{ij} q^r (1-q) Y_{ijr} \right]$$

The emergency facility u is handled as in the RPMP-EFC, described in Section 6.2.1; it has no fixed cost ($f_u = 0$).

The RFLP-EFC is formulated as follows:

(RFLP-EFC)

$$\text{minimize} \quad \alpha w_1 + (1-\alpha) w_2 \tag{6.24}$$

$$\text{subject to} \quad \sum_{j \in J} Y_{ijr} + \sum_{j \in NF} \sum_{s=0}^{r-1} Y_{ijs} = 1 \qquad \forall i \in I, r = 0, \ldots, m-1 \tag{6.25}$$

$$Y_{ijr} \leq X_j \qquad \forall i \in I, j \in J, r = 0, \ldots, m-1 \tag{6.26}$$

$$\sum_{r=0}^{P-1} Y_{ijr} \leq 1 \qquad \forall i \in I, j \in J \tag{6.27}$$

$$X_u = 1 \tag{6.28}$$

$$X_j \in \{0,1\} \quad \forall j \in J \tag{6.29}$$

$$Y_{ijr} \in \{0,1\} \quad \forall i \in I, j \in J, r = 0, \ldots, m-1 \tag{6.30}$$

The formulation is identical to that of RPMP-EFC except:

- Fixed costs are included in objective w_1
- Constraint (6.4) is omitted
- The "level" index r is extended to $m-1$ instead of $P-1$ in summations and constraint indices

Constraint (6.28) is not strictly necessary since facility u has 0 fixed cost, but including the constraint in the formulation tightens the Lagrangian relaxation. Note that Theorem 6.1 applies to the RFLP-EFC as well.

6.5.2 Solution Method

To solve (RFLP-EFC), we relax constraints (6.25) to obtain the following Lagrangian subproblem:

(RFLP-EFC-LR$_\lambda$)

$$\text{minimize} \qquad z(\lambda) = \sum_{j\in J} f_j X_j + \sum_{i\in I}\sum_{j\in J}\sum_{r=0}^{m-1} \tilde{\psi}_{ijr} + \sum_{i\in I}\sum_{r=0}^{m-1} \lambda_{ir} \tag{6.31}$$

$$\text{subject to} \qquad Y_{ijr} \le X_j \qquad \forall i \in I, j \in J, r = 0,\ldots,m-1 \tag{6.32}$$

$$\sum_{r=0}^{m-1} Y_{ijr} \le 1 \qquad \forall i \in I, j \in J \tag{6.33}$$

$$X_u = 1 \tag{6.34}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{6.35}$$

$$Y_{ijr} \in \{0,1\} \qquad \forall i \in I, j \in J, r = 0,\ldots,m-1 \tag{6.36}$$

In the objective function (6.31),

$$\tilde{\psi}_{ijr} = \begin{cases} \psi_{ijr} - \lambda_{ir}, & \text{if } j \in F \\ \psi_{ijr} - \lambda_{ir} - \left(\sum_{s=r+1}^{m-1} \lambda_{is}\right) = \psi_{ijr} - \sum_{s=r}^{m-1} \lambda_{is}, & \text{if } j \in NF \end{cases} \tag{6.37}$$

The benefit γ_j of opening facility j is computed as

$$\gamma_j = \alpha f_j + \sum_{i\in I} \min\left\{0, \min_{r=0,\ldots,m-1}\{\tilde{\psi}_{ijr}\}\right\}. \tag{6.38}$$

X_u is set to 1, and for $j \neq u$, X_j is set to 1 if $\gamma_j < 0$ (or if $\gamma_k \geq 0$ for all $k \in J$ but is smallest for j, since at least one facility in addition to u must be open in any feasible solution to (RFLP-EFC)); Y_{ijr} is set following the criteria described in Section 6.3.1.

At each Lagrangian iteration, we find an upper bound by opening the facilities that are open in the solution to (RFLP-EFC-LR$_\lambda$) and greedily assigning customers to them. In addition, we perform an "add" and a "drop" heuristic on each solution whose objective value is less than 1.2UB, where UB is the best known upper bound. The add (drop) heuristic considers opening (closing) facilities if doing so decreases the objective value. Each heuristic is performed until no further adds or drops will improve the solution. Then, for every fifth solution, the DC exchange heuristic is performed, as described in Section 6.3.2.

The subgradient optimization and branch-and-bound procedures are exactly as described for the RPMP-EFC, except that branch-and-bound nodes are fathomed if the lower bound at that node is within ϵ of the best known upper bound, if $|J|$ (rather than P) facilities have been forced open, or if $|J|-1$ (rather than $|J|-P$) facilities have been forced closed.

6.6 A Modification

In our preliminary computational testing, we found that the subgradient optimization procedure had difficultly converging to a tight lower bound for the RFLP-EFC. We believe the problem to lie in the large number of multipliers that must be updated (nm of them, as opposed to nP in the RPMP-EFC). To counteract this effect, we propose the following modification of our model and algorithm. Since the probability of many facilities failing simultaneously is small, ignoring the simultaneous failure of more than, say, 5 facilities may result in a very small loss of accuracy. At the same time, the reduction in the number of multipliers may result in a very large improvement in computational performance. Customers would only be assigned to facilities at levels 0 through 4, and higher-level assignments would not be included either in the objective function or in the constraints. In fact, if we interpret m as the number of levels to be assigned, rather than as the cardinality of J, then the objectives w_1 and w_2 and the formulation of (RFLP-EFC) remain intact under this new modeling scheme, as does the Lagrangian relaxation (RFLP-EFC-LR$_\lambda$) and the algorithm for solving it. The emergency facility may become irrelevant in this case, since it is generally used only when all open facilities have failed, but it may still play a role in the solution if the emergency cost is smaller than the cost of serving a given customer from, say, its fourth nearest facility when the first three have failed.

We observed similar convergence problems in the RPMP-EFC when P was large. The same modification may be made to (RPMP-EFC) by replacing P with m (except in constraint (6.4)). We have found this modification to be very effective for both problems; our computational experience with this modification is presented in Section 6.7.4.

6.7 Computational Results

6.7.1 Experimental Design

We tested our algorithms on a 25-node data set consisting of random data and the 49-node data set described by Daskin (1995). All nodes serve as both customers and potential facility locations. In the 25-node data set, demands are drawn from $U[0, 10^5]$ and rounded to the nearest integer; fixed costs (for the RFLP-EFC) are drawn from $U[4000, 8000]$ and rounded to the nearest integer. Latitudes and longitudes are drawn from $U[0, 1]$ and transportation costs are set equal to the Euclidean distance, per unit demand. Emergency costs θ_i are set to 10 for each customer, $q = 0.05$, and all facilities are failable. The 49-node data set represents the state capitals of the continental United States plus Washington, DC. Demands are equal to the state population and fixed costs are equal to the median home value, both from the 1990 census. Transportation costs are set equal to the great-circle distance times 10^{-5}, per unit demand. Emergency costs θ_i are set equal to 10^5, $q = 0.05$, and all facilities are failable. The emergency costs for both data sets are meant to model situations in which losing a customer is extremely costly.

We tested the RPMP-EFC algorithm on both data sets for several values of P, as well as the RFLP-EFC algorithm, using six different values of α. We executed the Lagrangian relaxation/branch-and-bound process to an optimality tolerance of 0.1%, or until 300 seconds (5 minutes) of CPU time had elapsed. The algorithm was tested on a Dell Inspiron 7500 notebook computer with a 500 MHz Pentium III processor and 128 MB memory. Parameter values for the Lagrangian relaxation algorithm are given in Table 6.1. The number of levels included in the objective function and constraints (m; see Section 6.6) was set to 5 except when $P < 5$, in which case m was set equal to P.

Table 6.1: Parameters for Lagrangian relaxation procedure.

Parameter	Value
Optimality tolerance (ϵ)	0.001
Maximum number of iterations ($n_{\max}$) at root node	1200
Maximum number of iterations ($n_{\max}$) at child nodes	600
Initial value of β	2
Number of non-improving iterations before halving β	30
Minimum value of β ($\beta_{\min}$)	10^{-8}
Initial value for λ_{is}	0

6.7.2 Algorithm Performance

Table 6.2 summarizes the results for the RPMP-EFC, Table 6.3 for the RFLP-EFC. The Overall LB, UB, and Gap columns give the lower and upper bounds and the percentage gap, while the columns marked Root LB, UB, and Gap give the lower and upper bounds and the gap at the root node. The column marked # Lag Iter gives the total number of Lagrangian iterations, # BB Nodes gives the total number of branch-and-bound nodes, and CPU Time gives the total number of CPU seconds required.

The algorithm produces tight bounds for the RPMP-EFC when P is small, and for the RFLP-EFC, usually finding the optimal solution without any branching. For larger values of P, the performance deteriorates somewhat, producing large root-node gaps in some cases. However, the lower bounds quickly increased at a relatively shallow depth in the branch-and-bound tree, suggesting that our initial multipliers may be poor for these problems but that good bounds can be obtained at child nodes once the multipliers have been improved. (It is generally desirable to set initial multipliers to something other than 0 in a Lagrangian relaxation algorithm, but we were unable to find a non-zero value that performed well for multiple instances of the data.) Even for the problems with the largest

Table 6.2: Algorithm results: RPMP-EFC.

# Nodes	P	α	Overall LB	Overall UB	Overall Gap	Root LB	Root UB	Root Gap	# Lag Iter	# BB Nodes	CPU Time
25	4	1.0	18689.1	18707.7	0.10%	18689.1	18707.7	0.10%	561	1	4.0
25	4	0.8	19019.4	19036.5	0.09%	19019.4	19036.5	0.09%	176	1	1.3
25	4	0.6	19346.2	19365.3	0.10%	19346.2	19365.3	0.10%	225	1	1.6
25	4	0.4	19675.4	19694.0	0.09%	19675.4	19694.0	0.09%	226	1	1.7
25	4	0.2	20003.4	20022.8	0.10%	20003.4	20022.8	0.10%	91	1	0.7
25	4	0.0	20338.9	20351.6	0.06%	20338.9	20351.6	0.06%	91	1	0.7
25	8	1.0	8846.3	8854.4	0.09%	8802.0	8854.4	0.60%	1929	5	13.7
25	8	0.8	9116.6	9125.4	0.10%	9116.6	9125.4	0.10%	868	1	6.2
25	8	0.6	9387.3	9396.4	0.10%	9387.3	9396.4	0.10%	588	1	4.3
25	8	0.4	9652.5	9662.1	0.10%	9652.5	9662.1	0.10%	584	1	4.2
25	8	0.2	9881.1	9889.9	0.09%	9881.1	9889.9	0.09%	462	1	3.4
25	8	0.0	10108.5	10117.7	0.09%	10108.5	10117.7	0.09%	281	1	2.1
25	12	1.0	4539.1	4543.3	0.09%	4398.6	4543.3	3.29%	3768	11	26.4
25	12	0.8	4737.9	4742.6	0.10%	4681.6	4742.6	1.30%	1941	5	13.7
25	12	0.6	4936.9	4941.8	0.10%	4570.2	4941.8	8.13%	2915	7	20.6
25	12	0.4	5136.2	5141.1	0.10%	5118.8	5141.1	0.44%	1378	3	9.8
25	12	0.2	5335.5	5340.4	0.09%	5330.3	5340.4	0.19%	1265	3	8.7
25	12	0.0	5534.2	5539.7	0.10%	5534.2	5539.7	0.10%	1034	1	7.3
49	5	1.0	502246.0	502732.0	0.10%	502246.0	502732.0	0.10%	643	1	10.3
49	5	0.8	517695.0	518210.0	0.10%	517695.0	518210.0	0.10%	338	1	5.8
49	5	0.6	533173.0	533687.0	0.10%	533173.0	533687.0	0.10%	352	1	6.1
49	5	0.4	547771.0	548279.0	0.09%	547771.0	548279.0	0.09%	426	1	7.3
49	5	0.2	561886.0	562437.0	0.10%	561886.0	562437.0	0.10%	340	1	5.8
49	5	0.0	575593.0	576153.0	0.10%	575593.0	576153.0	0.10%	319	1	5.3
49	10	1.0	275428.0	275701.0	0.10%	260337.0	275701.0	5.90%	3034	7	50.9
49	10	0.8	283343.0	283601.0	0.09%	283343.0	283601.0	0.09%	713	1	12.2
49	10	0.6	291212.0	291501.0	0.10%	291212.0	291501.0	0.10%	828	1	14.4
49	10	0.4	299110.0	299402.0	0.10%	299110.0	299402.0	0.10%	1061	1	18.1
49	10	0.2	307008.0	307302.0	0.10%	306659.0	307302.0	0.21%	1262	3	21.8
49	10	0.0	314904.0	315202.0	0.09%	314904.0	315202.0	0.09%	703	1	12.5
49	20	1.0	111580.0	113330.0	1.57%	67665.3	113330.0	67.49%	15872	42	309.8
49	20	0.8	119544.0	119663.0	0.10%	107651.0	119663.0	11.16%	14797	37	278.0
49	20	0.6	125719.0	125995.0	0.22%	105994.0	125995.0	18.87%	14373	31	300.4
49	20	0.4	132196.0	132328.0	0.10%	125972.0	132328.0	5.05%	9005	23	178.6
49	20	0.2	138522.0	138661.0	0.10%	134446.0	138661.0	3.14%	3075	7	72.2
49	20	0.0	144783.0	144926.0	0.10%	89001.2	144926.0	62.84%	11320	29	230.0

Table 6.3: Algorithm results: RFLP-EFC.

# Nodes	α	Overall LB	Overall UB	Overall Gap	Root LB	Root UB	Root Gap	# Lag Iter	# BB Nodes	CPU Time
25	1.0	39650.9	39684.10	0.08%	39650.9	39684.10	0.08%	231	1	1.8
25	0.8	36070.8	36106.20	0.10%	36070.8	36106.20	0.10%	203	1	1.6
25	0.6	32361.3	32392.00	0.09%	32361.3	32392.00	0.09%	142	1	1.2
25	0.4	27051.7	27076.30	0.09%	27051.7	27076.30	0.09%	194	1	1.7
25	0.2	18599.0	18617.40	0.10%	18599.0	18617.40	0.10%	402	1	3.4
25	0.0	903.2	904.04	0.09%	903.2	904.04	0.09%	776	1	5.3
49	1.0	855960.0	856810.00	0.10%	855786.0	856810.00	0.12%	1312	3	22.5
49	0.8	790227.0	791014.00	0.10%	790227.0	791014.00	0.10%	181	1	4.5
49	0.6	707278.0	707982.00	0.10%	707278.0	707982.00	0.10%	382	1	8.5
49	0.4	589108.0	589677.00	0.10%	589108.0	589677.00	0.10%	544	1	13.3
49	0.2	404501.0	404903.00	0.10%	404016.0	404903.00	0.22%	1416	3	25.2
49	0.0	19285.9	19302.80	0.09%	17635.5	19302.80	9.45%	2609	7	41.4

gaps, the branch-and-bound algorithm was very successful, solving the problem to 0.1% optimality within 5 minutes for all but two problems, and yielding gaps less than 2% for those problems. One surprising aspect of the results is that the algorithm often performs worse for $\alpha = 1$ than for smaller α. We believe this is because $\alpha = 1$ represents a pure P-median problem with many extraneous variables and constraints; the extra variables have no bearing on the objective function, leading to a large number of optimal solutions that are difficult to prove optimal. Location problems with highly regular cost structures (e.g., many customers are equidistant from many facilities) are well known to be difficult to solve.

6.7.3 Tradeoff Curves

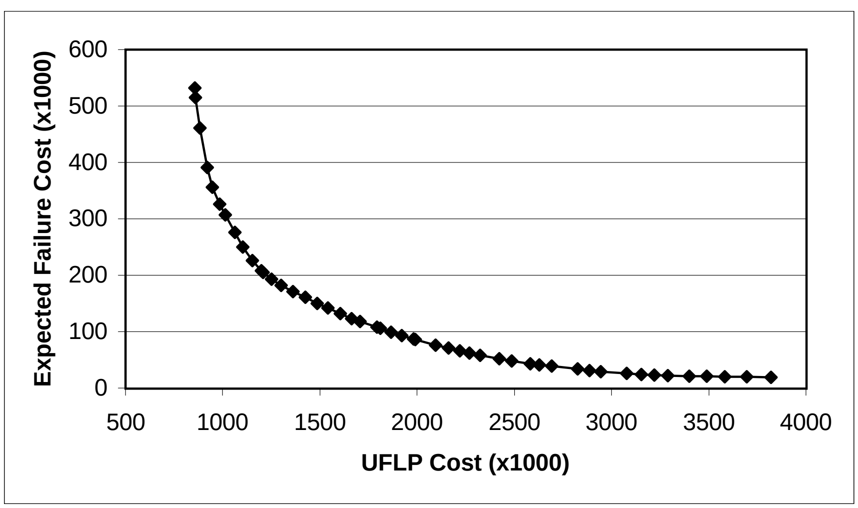
We constructed the tradeoff curve for the RFLP-EFC using the 49-node data set as described in Section 6.4; the results are pictured in Figure 6.1. The horizontal axis plots the UFLP cost (objective 1) while the vertical axis plots the expected failure cost (objective 2). Each point on the curve represents a different solution; the optimal UFLP solution (found by solving (RFLP-EFC) with $\alpha = 1$) is the left-most point on the curve. The steepness of the left part of the curve indicates that large improvements in reliability can be attained without large increases in UFLP cost. The 10 left-most solutions on the curve are listed in Table 6.4, along with their relationship to the UFLP solution and the number of facilities open in the solution. Decision makers may be reluctant to undertake large increases in UFLP cost, but they may be willing, for example, to expend 7% more to reduce expected failure cost by 27%, as in solution 4. The points at the right of the

Table 6.4: First 10 solutions in curve: RFLP-EFC.

Soln #	Obj 1	Obj 2	% Increase Obj 1	% Decrease Obj 2	# Locations
1	856810	532199	0.0%	0.0%	6
2	860078	514758	0.4%	3.3%	6
3	883656	460699	3.1%	13.4%	7
4	919203	391149	7.3%	26.5%	8
5	946914	356139	10.5%	33.1%	9
6	984969	326149	15.0%	38.7%	10
7	1014350	306754	18.4%	42.4%	11
8	1062410	275649	24.0%	48.2%	12
9	1104380	250493	28.9%	52.9%	13
10	1151970	226437	34.4%	57.5%	14

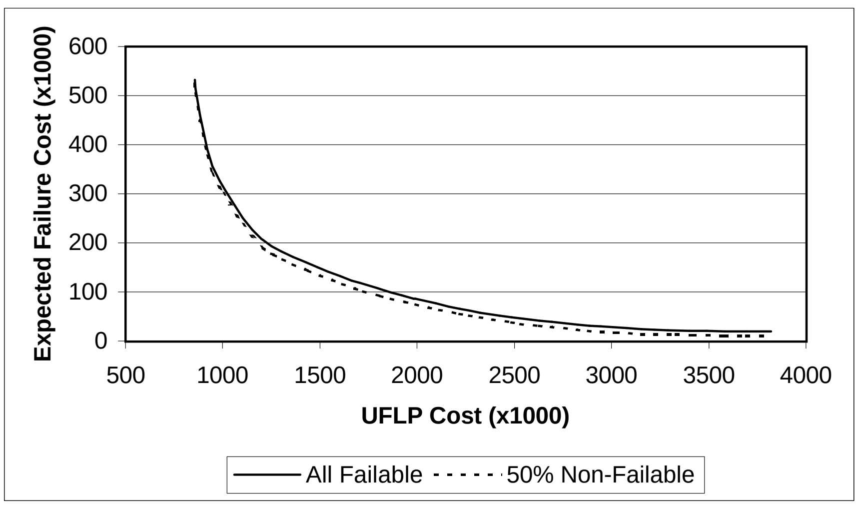
tradeoff curve are unlikely to be of much interest, as they represent solutions in which nearly all of the facilities are open; they have extremely low failure costs but are very expensive to implement. In general, we find that the left portion of the tradeoff curve is quite steep.

Figure 6.1: RFLP-EFC tradeoff curve for 49-node data set.



The tradeoff curve in Figure 6.1 was constructed assuming all facilities are failable. Firms may be interested in knowing how the tradeoff curve changes if some of the facilities in J are non-failable. This may help firms make decisions about which contracts should be shored up or which DCs should be owned in-house, for example. Figure 6.2 contains two tradeoff curves, the one in which all facilities are failable (discussed earlier) and another in which 25 of the 49 facilities, chosen randomly, are designated as non-failable. The inclusion of non-failable facilities has the effect of shifting the tradeoff curve favorably toward the origin. (In addition, problems with more non-failable facilities generally produce tighter bounds and require less computation time.)

Figure 6.2: Shifting tradeoff curve.

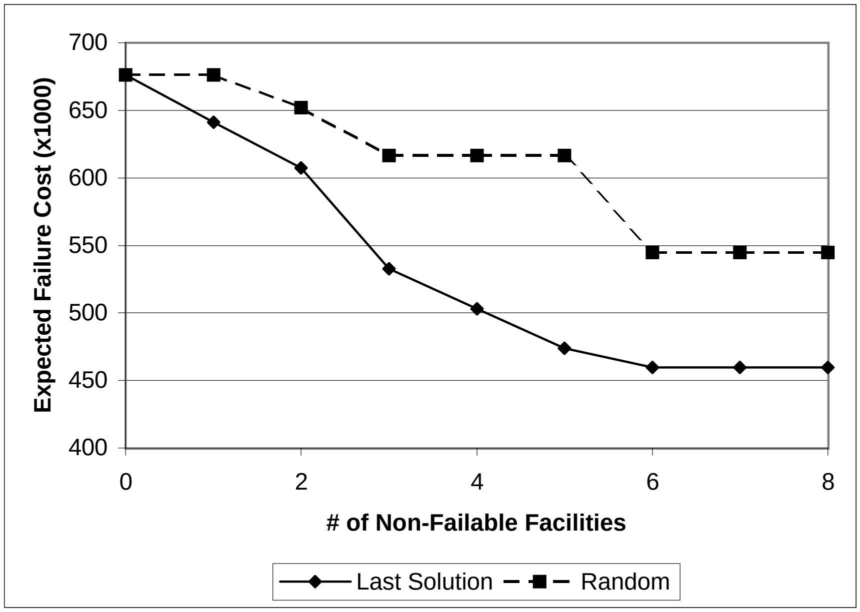


A natural question to ask is, how many non-failable facilities need to be included in the set of potential facility sites to achieve a satisfactory level of reliability? Figure 6.3 addresses this question by plotting the decrease in objective function value as the number of non-failable facilities increases, using the RFLP-EFC and the 49-node data set with $\alpha = 0.8$ and $q = 0.25$. In the lower curve (marked "Last Solution"), the non-failable facilities were selected from the previous solution. This curve represents how the objective function might change if the firm could make any facilities that it likes non-failable. In the upper curve (marked "Random"), the non-failable facilities are chosen randomly. This represents the case in which the firm has no control over which facilities are non-failable. Note that in both cases, the horizontal axis plots the number of non-failable facilities that are in J; not all of these will necessarily be chosen in the solutions. From the chart it is apparent that only a few non-failable DCs are necessary to make the solution as a whole substantially more reliable. For our client, the durable goods manufacturer, this means that the company needs to own only a few DCs in-house to make the system perform well when third-party distributors fail. It also allows the firm to threaten to cancel contracts with badly performing distributors, since the firm can credibly claim to be able to operate without the distributor, at least in the short term, while they establish a contract with a new distributor. This is an important issue in contract negotiation and enforcement.

6.7.4 Number of Levels

In this section we briefly explore the impact of changing the number of levels, m, as discussed in Section 6.6. We solved the RPMP-EFC with $P = 10$ and the RFLP-EFC using the 49-node data set, $\alpha = 0.8$, and $q = 0.05$, testing various values of m. The results are presented in Table 6.5. (Note that the last entry for each problem corresponds to the case in which we do not use the modification suggested in Section 6.6. $m = 50$ for the RFLP-EFC and $m = 11$ for the RPMP-EFC since $|J|$ and P increased by 1 when the emergency facility was added.) The column marked Time/Iter gives the average number of CPU seconds spent on each Lagrangian iteration. From the table it

Figure 6.3: Changing the number of non-failing facilities.



is apparent that using larger values of m generally results in larger root-node optimality gaps, more Lagrangian iterations, more branch-and-bound nodes, and more time spent on each iteration (due to loops of the form "for $r = 0, \ldots, m-1$"). It is worth pointing out that in all cases, regardless of the value of m, the algorithm returned the same set of locations, indicating that the computational improvement came at no loss of solution accuracy, though of course we cannot prove that this will hold in general.

6.8 Chapter Summary

In this chapter we presented expected failure cost (EFC) reliability models. Unlike the maximum failure cost (MFC) models, the EFC models assume that multiple facilities can fail simultaneously. We formulated models based on the P-median problem and the

Table 6.5: Sensitivity to m.

Problem	m	Root Gap	# Lag Iter	# BB Nodes	CPU Time	Time/Iter
RPMP-EFC	3	0.10%	285	1	4.0	0.01
RPMP-EFC	5	0.09%	713	1	12.2	0.02
RPMP-EFC	7	0.29%	1294	3	28.1	0.02
RPMP-EFC	9	0.13%	1253	3	32.9	0.03
RPMP-EFC	11	0.30%	1868	5	56.7	0.03
RFLP-EFC	3	0.08%	114	1	2.3	0.02
RFLP-EFC	5	0.10%	181	1	4.3	0.02
RFLP-EFC	8	0.75%	3845	11	157.4	0.04
RFLP-EFC	12	0.94%	3927	11	197.0	0.05
RFLP-EFC	20	0.73%	1637	3	87.4	0.05
RFLP-EFC	50	0.96%	2430	4	333.7	0.14

uncapacitated fixed-charge location problem, called the RPMP-EFC and RFLP-EFC, respectively. Like the MFC models, the EFC models make use of “backup” assignments, but in this case multiple levels of backups are required. In both models, the expected transportation cost, taking into account the costs that result from facility failures, is included in the objective function. The tradeoff of interest is between the operating cost (the traditional PMP or UFLP objective function) and the expected failure cost. Tradeoff curves can be generated using the weighting method of multi-objective programming. Both models are solved using Lagrangian relaxation, with promising results. We demonstrated empirically that the interesting portion of the tradeoff curve is steep, indicating that reliability can be drastically improved without large increases in operating costs. This is a critical issue for decision-makers who may be reluctant to expend greater sums *for sure* in order to hedge against *possible* failures in the future.

For large values of P in the RPMP-EFC, and for the RFLP-EFC, straightforward application of our algorithm yielded large bounds at the root node. We proposed a modification that entails assigning facilities only to a pre-specified level m (we used

$m = 5$). This modification tightens these bounds considerably with little or no loss of accuracy. In our computational tests, we found that the choice of m has a large impact on computational performance but *no* impact on the solution found. For different values of m, the objective function for the solutions differed slightly since higher-level terms are excluded for smaller values of m. However, we found this difference to be less than 0.02% in all cases, and less than 0.0005% when $m \geq 5$. This addresses an important question about the bounds produced by our algorithm. In particular, when a Lagrangian relaxation algorithm produces lower bounds that are loose, one always wants to know whether this is the theoretical lower bound or simply a practical lower bound that might be improved by a different multiplier updating method or different choices of algorithm parameters. Consider the last entry in Table 6.5. When we began testing, we assumed that the theoretical bound for this problem was 0.96% away from the optimal solution, or close to it. When $m = 3$, however, we get a lower bound that is only 0.08% from the upper bound, and since this upper bound is very close (within 0.02%, as discussed above) of the upper bound when all assignment levels are included in the objective function, we can be confident that the theoretical lower bound is no more than 0.1% from the optimal solution. This suggests that the size of the practical bounds is to some extent determined by our implementation of the multiplier updating routine, and not by the theoretical bound, and that we might tighten this bound even further by improving this routine. (This is especially important for the larger problems tested, which resulted in bounds significantly larger than 0.1% at the root node.)

We also note that our upper-bound heuristic and improvement routines are highly effective, yielding the optimal solution at the root node in all cases tested, generally finding it within the first 100 iterations or so. This suggests that very good solutions can be found very quickly, if a theoretical guarantee of optimality is not required.

Clearly, the main drawback of our models is the assumption that failable facilities all have the same probability q of failing. This assumption is necessary to allow us to compute the probability that a customer is served by its level-r facility without explicitly knowing its lower-level assignments, only that there are r of them and that they are failable. Increasing the number of probabilities results in an exponential increase in the number of terms in the objective function, since one term is required for each possible combination of the failure probabilities of the r lower-level assignments. We intend to study this issue in future research to find an objective function that can accommodate multiple failure probabilities. Another simplifying assumption we made is that failures are statistically independent of one another. This assumption may be inaccurate—weather- and labor-related failures may be dependent on those of nearby facilities—but is necessary for tractability. Again, future research may identify ways to incorporate dependence into the EFC models.

Finally, we note that if decision makers are interested only in total expected cost, not in the tradeoff between the PMP or UFLP objective and the expected failure cost, the two objectives can easily be replaced with a single objective representing the expected cost. For the RPMP-EFC, this simply means setting $\alpha = 0$. For the RFLP-EFC, one would add the fixed cost term to w_2 and then set $\alpha = 0$. In either case, the solution method remains the same. Some decision makers may prefer these formulations as they address the common objective of minimizing long-run cost. We have chosen to formulate the problems as we did because the multi-objective framework provides greater flexibility; more importantly, it allows us to demonstrate, via tradeoff curves, the large improvements in reliability that are possible with only small increases in the objectives under which firms have historically evaluated facility location decisions.

Chapter 7

Conclusions and Future Research

In this dissertation, we presented models for *robust* and *reliable* supply chain design. Robustness refers to the ability of a solution to perform well under various realizations of the random parameters, while reliability refers to the ability of a solution to perform well even when parts of the system fail. Robustness is a measure that has been studied widely in the operations research literature. Various measures of robustness have been considered; in this dissertation, we consider minimizing the expected cost, and in some cases adding a bound on the regret in any scenario. Our robustness models are based on the location model with risk pooling (LMRP). Reliability, on the other hand, has received relatively little attention, except in limited contexts. We propose models for reliable facility location, based on the classical P-median problem (PMP) and uncapacitated fixed-charge location problem (UFLP). These models attempt to find solutions that are both inexpensive and reliable, and we have shown empirically that large improvements in reliability are often possible with little additional cost.

Our solution methods for both the stochastic LMRP studied in Chapter 3 and the expected failure cost reliability models studied in Chapter 6 performed well, producing consistently tight bounds and short computation times. We were less successful in solving the p-robust optimization models presented in Chapter 4 and the maximum failure cost reliability models presented in Chapter 5. These models have similar structures, in that they all have the PMP or UFLP as an underlying model, plus a set of constraints for each scenario or facility that requires some cost, related to but not equal to the objective value, to be less than a given constant. The objective values of the continuous relaxations of all of these models seem to increase very slowly as the right-hand side of the p-robustness or reliability constraints is decreased. This makes solving these problems by Lagrangian relaxation very difficult, since the IP objective values increase much more sharply as the constraints are tightened. We intend to study the continuous relaxations of these models further to explain why this curious behavior occurs and to develop alternative models or solution methods that circumvent the problem. The p-SLMRP seems to be a good candidate for Lagrangian methods, assuming that the bounds can be tightened. However, Lagrangian relaxation seems less effective, or at least less consistent, for the maximum failure cost reliability models, suggesting that other IP methods may be needed. (Benders decomposition seems a promising avenue to explore.)

Another important open issue stemming from this research is Conjecture 4.1, which addresses the relationship between the infeasibility of the continuous relaxation of the p-robust problems and the unboundedness of their Lagrangian relaxations. We can prove this conjecture for the special case of the p-robust UFLP, but we hope to prove it for the more general case, as well.

All of our models are extensions of NP-hard problems. It may be instructive to study similar robust and reliable extensions of polynomially solvable problems (for example, median problems on specially structured networks). One of the drawbacks of the popular minimax regret robustness measure is that some easy problems (like the shortest path problem) have robust versions that are NP-hard. We would like to study whether our measures preserve the "easiness" of these problems.

The reliability models in Chapters 5 and 6 represent a new direction in supply chain design under uncertainty. We would like to use the ideas studied in these chapters to formulate and solve reliability models based on more sophisticated supply chain design models like the LMRP, rather than facility location problems like the PMP and UFLP. Reliability formulations of the LMRP would be much more difficult to solve, both because of the non-linearities and because the square-root function ties together terms that are separable in linear formulations. Nevertheless, such models would be an important contribution to the literature on reliable supply chain design. We would also like to study formulations of the expected failure cost models that allow failable facilities to have different failure probabilities, possibly allowing dependence among them. Finally, we intend to explore other supply chain and logistics problems to which the reliability concept can be applied, for example scheduling, inventory policies, and transportation.

Appendix A

Counterexample to p-Robust ISP Algorithm

Gutiérrez and Kouvelis's (1995) paper on the international sourcing problem (ISP) essentially provides an algorithm for solving a p-robust version of the UFLP, since the ISP can be reduced to the UFLP. The algorithm takes p and N (an integer) as inputs and returns the N "most robust" solutions, or all p-robust solutions if there are fewer than N of them. "Most robust" means having minimum max regret across all scenarios. The authors claim that "...when the algorithm finishes executing, for a given pre-specified robustness parameter p, it will either have identified the best N robust solutions, or if it identifies only $n < N$, possibly $n = 0$ robust solutions, then we can guarantee that these are the only robust solutions for the given p." (p. 184) We dispute this claim.

The algorithm maintains a separate branch-and-bound tree for each scenario, and all trees are branched and fathomed simultaneously so they all have the same structure at the same time. Lower bounds are obtained at each node by summing the linking constraints across the customers[1] and relaxing the integrality constraints. The solution to the resulting "weak relaxation" provides a lower bound, and if it happens to be integer, it provides an upper bound as well. Nodes are fathomed for three reasons:

1. When the weak relaxation is infeasible. This happens when all facilities eligible to serve a given customer are fixed closed.

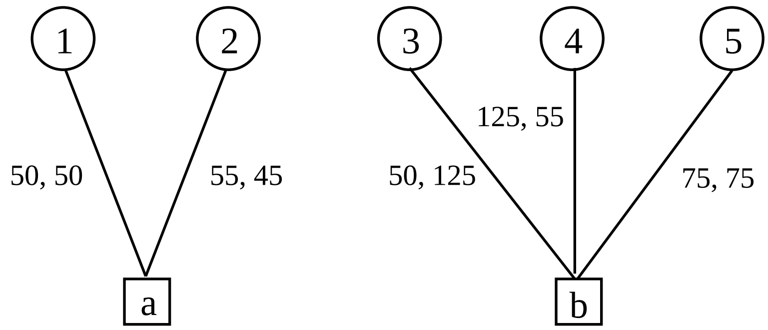
2. When the lower bound for a single-scenario problem is greater than $(1 + p)$ times the optimal objective value for that scenario (in which case searching that portion of the branch-and-bound tree cannot produce a p-robust solution).

3. When p is reduced because N p-robust solutions have been found. In this case, nodes corresponding to the "extra" solutions are fathomed. (See the last few lines of the algorithm, on p. 184.)

The problem is with reason #3 above. The authors implicitly assume that a solution found at the child of a branch-and-bound node cannot have a smaller maximum regret than the solution found at the node itself, but this is not true. The child node will certainly have worse regret for the scenario in question but may reduce the regret for the other scenarios, thereby reducing the maximum regret. Furthermore, when a node is fathomed from one scenario-tree, the corresponding nodes are fathomed from all scenario-trees. This means that if two scenarios generate feasible solutions at a given node and

[1] In the ISP, "customers" and "facilities" are replaced by "factories" and "suppliers," respectively. We will continue to use the UFLP terminology.

Figure A.1: ISP example.



one is good but the other is bad, we throw away the good with the bad by fathoming.

Consider the following example. There are 5 facility locations, 2 customers, and 2 scenarios. Not all facilities are eligible to serve all customers. Figure A.1 shows the facilities, customers, and the associated data. The numbers next to the links give scenario-specific transportation costs (scenario 1, then scenario 2). All facilities have fixed costs of 50, and both customers have demand of 1. There are no minimum procurement requirements (of the type described in Section 3.1 of Gutiérrez and Kouvelis's paper).

Let $p = 1$ and $N = 1$ (that is, find the single solution with minimum max regret, and start with $p = 1$). By inspection one can confirm that the optimal scenario solutions are $Y_1^* = (1, 0, 1, 0, 0)$ for scenario 1 with objective value $Z_1(Y_1^*) = 200$ and $Y_2^* = (0, 1, 0, 1, 0)$ with $Z_2(Y_2^*) = 200$ for scenario 2.[2] The minimax regret solution is $Y = (1, 0, 0, 0, 1)$ with regrets $R_1 = R_2 = 0.125$ (where R_i is the percent regret if scenario i occurs). Since each facility is eligible to serve only a single customer, the weak relaxation solved at each node of the branch-and-bound trees is equivalent to the LP relaxation of the UFLP.

[2]Here we use the notation from Gutiérrez and Kouvelis's paper. Y represents a location vector and Z represents a cost.

Furthermore, these LP relaxations happen to have integer solutions, so one can confirm the optimal solutions and objective values given below by inspection.

The branch-and-bound trees are shown in Figure A.2, with (single-scenario) objective value, solution vector, and regret displayed next to the nodes. We now walk through the algorithm step by step.

Step 0: Solve the root nodes (with no variables fixed). The optimal solution for scenario 1 is to locate at 1 and 3, with cost 200 and regret $R_1 = 0$ if scenario 1 occurs and $R_2 = 0.375$ since the cost of this solution if scenario 2 occurs is 275. Similarly, the optimal solution for scenario 2 is to locate at 2 and 4. This solution has cost 200 and regret $R_1 = 0.4$ (since the solution costs 280 if scenario 1 occurs) and $R_2 = 0$.

Step 1: Choose a scenario, node, and variable to branch on. We'll choose scenario 1, node 1, and the variable y_1. (Note that these choices are consistent with the branching rules described on p. 182 of Gutiérrez and Kouvelis's paper.) We remove node $k = 1$ from both trees (i.e., we no longer consider these nodes for branching) and create nodes $1_s^{[0]}$ and $1_s^{[1]}$ with y_1 fixed to 0 and 1, respectively. Since both problems are feasible, at the end of this step $L_{\text{New}} = \{1_1^{[0]}, 1_1^{[1]}, 1_2^{[0]}, 1_2^{[1]}\}$ and we go to step 2. (L_{New} is the list of new nodes whose weak relaxations are feasible. The notation $1_2^{[0]}$ means child node [0] of node 1, scenario 2.)

Step 2: The optimal solution at node $1_1^{[0]}$ (child 0 for scenario 1) is $y = (0, 1, 1, 0, 0)$ with objective value $z = 205$ and regret $R_1 = 0.025, R_2 = 0.35$. For scenario 2, the optimal solution at the root node already had $y_1 = 0$, so this solution remains optimal for the child node. Both solutions pass the lower-bound robustness test since the lower bounds are within p of the optimal solution for the scenario.

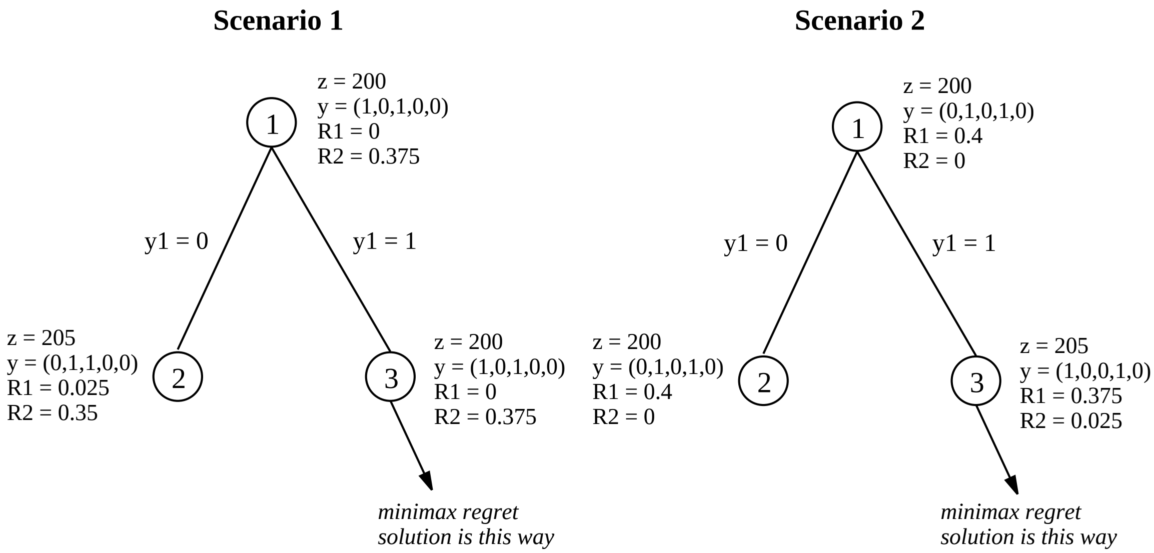
Step 3: The optimal solution for scenario 1 is the same as at the root node since this solution already has $y_1 = 1$. For scenario 2, the optimal solution is $y = (1, 0, 0, 1, 0)$ with $z = 205$ and regret $R_1 = 0.375, R_2 = 0.025$. Both solutions pass the lower-bound robustness test. All nodes are added to their respective trees, and since all solutions are integral, $L_{\text{Int}} = L_{\text{New}}$. ($L_{\text{Int}}$ is the list of integer solutions found at the current iteration.)

Step 4: The maximum regret for all solutions across all scenarios is less than p, so we don't remove any nodes from L_{Int} and we go to step 5.

Step 5: The maximum regret is less than p (=1) for all solutions, so $L_R = L_{\text{Int}}$. (L_R is the list of p-robust solutions for the current value of p.) Since $|L_R| = 4$ and $N = 1$, we need to drop 3 solutions from the list. The best solution is at node $1_1^{[0]}$ with maximum regret 0.35, so we drop nodes $1_1^{[1]}$, $1_2^{[0]}$, and $1_2^{[1]}$.

At this point, the algorithm terminates because all nodes have been fathomed. This causes two problems. First, when we drop nodes $1_1^{[1]}$ and $1_2^{[1]}$, we fathom the section of the tree that contains the optimal (minimax regret) solution. Second, when we fathom $1_2^{[0]}$, we must also fathom $1_1^{[0]}$ since we fathom nodes from all scenario-trees simultaneously. But this means fathoming at our current best solution, even though its child nodes may have better solutions.

Figure A.2: Branch-and-bound trees for ISP algorithm.



This example shows that the fathoming rules are incorrect; by fathoming, the algorithm often cuts off improving branches in the search tree. If the algorithm were modified so that nodes are not fathomed in step 5, it would probably require much more branching and much larger computation times than those reported by Gutiérrez and Kouvelis.

Appendix B

The Multiple-Choice Knapsack Problem (MCKP)

The multiple-choice knapsack problem (MCKP), introduced by Nauss (1978a) and Sinha and Zoltners (1979), is a variation of the classical knapsack problem (KP) in which the items are partitioned into classes, and exactly one item must be chosen from each class to minimize the objective function while obeying a single knapsack constraint. The problem can be formulated as follows:

$$\text{(MCKP)} \quad \text{minimize} \quad \sum_{k=1}^{m} \sum_{j \in N_k} c_{kj} x_{kj} \tag{B.1}$$

$$\text{subject to} \quad \sum_{j \in N_k} x_{kj} = 1 \qquad \forall k = 1, \ldots, m \tag{B.2}$$

$$\sum_{k=1}^{m} \sum_{j \in N_k} a_{kj} x_{kj} \leq b \tag{B.3}$$

$$x_{kj} \in \{0, 1\} \qquad \forall j \in N_k, k = 1, \ldots, m \tag{B.4}$$

The classes $N_k,\ k = 1, \ldots, m$, are mutually exclusive. The KP is often described in terms of packing a knapsack, say for a camping trip. One wants to maximize the value (according to some scale) of the items chosen while making sure the items can fit into the knapsack. The MCKP, then, is the problem of choosing one each from a number of item types: one flashlight, one map, one bag of trail mix, and so on. The name of the problem refers to the fact that within each class, we must choose a single option from among a set of items, like a multiple-choice exam.

The KP can be reduced to the MCKP by placing each item in a class with a copy of itself. The item has objective function and constraint coefficients equal to those from the KP; the copy has objective function and constraint coefficients equal to 0. The 0–1 decision for each item in the KP translates to a multiple-choice decision for each class in the MCKP. Since the KP is NP-hard, so is the MCKP. Like the KP, good algorithms have been published for the MCKP.

Most papers about the MCKP assume that $c_{kj} \geq 0,\ a_{kj} \geq 0$ for all $j \in N_k,\ k = 1, \ldots, m$. However, any problem instance can be transformed into one with non-negative coefficients as follows. Let

$$c^- = \left| \min \left\{ 0, \min_{j \in N_k, k=1,\ldots,m} \{c_{kj}\} \right\} \right|$$
$$a^- = \left| \min \left\{ 0, \min_{j \in N_k, k=1,\ldots,m} \{a_{kj}\} \right\} \right|$$

Transform the coefficients by adding c^- to each c_{kj} and a^- to each a_{kj}; also add ma^- to b. Once the problem has been solved, subtract mc^- from the objective function.

In addition, some papers formulate constraint (B.3) as a $\geq$ constraint instead of as a $\leq$ constraint. Once again, any instance that uses a $\leq$ constraint as in (B.3) can be converted into an equivalent instance that uses a $\geq$ constraint so it can be solved by an algorithm requiring that form. This is done by replacing a_{kj} with $a^+ - a_{kj}$ and b with $ma^+ - b$, where

$$a^+ = \max\left\{\frac{b}{m}, \max_{j\in N_k, k=1,\ldots,m}\{a_{kj}\}\right\}.$$

Since the subproblems discussed in this dissertation use $\leq$ constraints and may contain negative coefficients, both of these transformations may be necessary, depending on the algorithm chosen.

Sinha and Zoltners (1979) present an algorithm for solving the LP relaxation of (MCKP) and a branch-and-bound algorithm in which the LP relaxation can be efficiently re-optimized at child nodes. (Sinha and Zoltners use the $\geq$ form of constraint (B.3), and their results are stated here assuming that form.) They prove that if $c_{kr} < c_{ks}$ and $a_{kr} > a_{ks}$ for $r, s \in N_k$, then $x_{ks} = 0$ in every optimal solution to (MCKP), since item s is both cheaper and larger than item r; item r is said to be "integer dominated" by item s and may be omitted from the problem at the outset. (Sinha and Zoltners assert that for randomly generated problems with 50 or more variables per class, the expected number of integer-dominated variables is more than 90%; we have found similar results empirically in our computational tests.) If $c_{kr} < c_{ks} < c_{kt}$, $a_{kr} < a_{ks} < a_{kt}$, and $(c_{ks} - c_{kr})/(a_{ks} - a_{kr}) > (c_{kt} - c_{ks})/(a_{kt} - a_{ks})$ for $r, s, t \in N_k$, then $x_{ks} = 0$ in every optimal solution to the LP relaxation of (MCKP); such variables are called "LP-dominated." At the outset of Sinha and Zoltners's algorithm, the variables in each class are sorted in increasing order of objective coefficients and integer- and LP-dominated variables are omitted. Note that while integer-dominated variables may be omitted permanently, variables that are LP-dominated at the root node of the branch-and-bound tree may not be dominated at child nodes, and vice-versa. The key result underlying Sinha and Zoltners's algorithm is as follows: the optimal solution to the LP relaxation of (MCKP) either is all integer or has exactly two fractional variables, in which case the fractional variables are adjacent variables (after sorting) in the same class. Their algorithm begins by setting $x_{kj} = 1$ for the item with the smallest objective coefficient in each class. If this solution is feasible, it is optimal. If not, the algorithm identifies the class whose currently chosen variable can be swapped for the next (sorted) variable in its class with a minimum ratio of objective function coefficient to constraint coefficient; the algorithm proceeds in this manner until the knapsack constraint is satisfied. The last swap made is typically a "fractional" one.

Armstrong et al. (1983) introduce an algorithm that is the inverse of Sinha and Zoltners's algorithm in that it initially chooses the most expensive item in each class and progressively swaps items for cheaper ones until making any swap would violate the knapsack constraint. Sinha and Zoltners's algorithm is an "optimistic" one that maintains optimality while working toward feasibility; Armstrong's algorithm is a "pessimistic" one that maintains feasibility while working toward optimality. Armstrong et al. show that both algorithms are special cases of the dual simplex method; their advantage lies in the fact that only one non-basic variable from each class must be evaluated when choosing an entering variable. They embed both algorithms into a branch-and-bound method that efficiently re-optimizes the LP relaxation at the child nodes, choosing the optimistic or pessimistic algorithm based on which variables are forced to 0 at each branch.

At least two other variations of Sinha and Zoltners's algorithm have been published. Pisinger (1995) identifies a "core" set of classes that receive more algorithmic attention than the others. He proves certain minimality properties about his algorithm with respect to the size of the core and the amount of sorting required. His algorithm is faster than Sinha and Zoltners's algorithm but it is also significantly more complicated to implement, and moreover, it only applies to problems with integer coefficients and is therefore of less interest for our problem. Nakagawa et al. (2001) make a variable substitution that converts the LP relaxation of the MCKP into that of the KP, which is easier to solve. They prove theoretically that their bound is tighter than the bound from the LP relaxation of (MCKP), but their computational results show an improvement on the order of 0.0001%, making it not worth the extra coding.

Aggarwal, Deo, and Sarkar (1992) describe a Lagrangian relaxation-based algorithm for the MCKP. They relax the single knapsack constraint (B.3) to obtain a simple "multiple-choice problem," which can be solved efficiently for a given Lagrange multiplier λ. They present a polynomial-time algorithm for finding an optimal multiplier λ^*, then close any resulting optimality gap using branch-and-bound. The key feature of their algorithm is that λ^* is used throughout the branch-and-bound tree; the Lagrangian problem does not need to be re-solved at each child node. A large number (tens of thousands) of branch-and-bound nodes may be required, but each one can be processed extremely quickly.

Of the algorithms discussed here, the most promising two are the LP algorithm of

Armstrong et al. and the Lagrangian algorithm of Aggarwal, Deo, and Sarkar, as these are both simple to implement and perform well. After implementing and experimenting with both, we found that while the Lagrangian algorithm may outperform the LP algorithm for some problems, the variability in run times for this algorithm was very large, making it unappealing as an algorithm to solve Lagrangian subproblems. Moreover, while the Lagrangian algorithm produces both lower and upper bounds, a solution attaining the resulting lower bound cannot readily be obtained. As discussed in Section 4.4.1, such a solution is desirable unless the problem is solved to optimality, an impractical option since the run times are so variable. For both of these reasons, we have elected to use the LP algorithm in our computational testing. To obtain a lower-bound solution from this algorithm, one simply keeps track of both the best lower bound *and* the solution that produced it. This is the solution to a "restricted" LP relaxation of the MCKP in which some variables have been forced to 0.

We make one other change to Armstrong's algorithm. If the LP relaxation at a given node results in a fractional solution, then exactly two variables are fractional, and they are in the same class. Call the variables x_{kj} and $x_{k,j+1}$; they must be adjacent with respect to the sort order, and since Armstrong et al. use $\geq$ knapsack constraints, $c_{kj} \leq c_{k,j+1}$ and $a_{kj} \leq a_{k,j+1}$. An "easy" feasible solution can be obtained by setting $x_{kj} = 0$ and $x_{k,j+1} = 1$. Armstrong, et al. point out that there may be LP-dominated variables between the two fractional variables (with respect to the sort order), and that if one of these has a large enough constraint coefficient, setting it to 1 will produce a better feasible solution. In fact, though, *any* of the classes may contain a (possibly LP-dominated) variable such that if that variable is set to 1, the current variable in that class is set to 0, x_{kj} is set to 1, and $x_{k,j+1}$ is set to 0, the resulting solution is feasible and is cheaper than the "easy" solution. The search for such variables can be performed efficiently since one only needs to examine the variables between the current variable and the *first* variable whose constraint coefficient is large enough to satisfy the knapsack constraint. We have found this modification to yield better solutions than Armstrong's method in up to 70% of the branch-and-bound nodes, with an average improvement of up to 2.8% in the upper bound at a given node.

Bibliography

[1] Aggarwal, Vijay, Narsingh Deo, and Dilip Sarkar. 1992. The knapsack problem with disjoint multiple-choice constraints. *Naval Research Logistics* 39, no. 2: 213-227.

[2] Akinc, Umit and Basheer M. Khumawala. 1977. An efficient branch and bound algorithm for the capacitated warehouse location problem. *Management Science* 23, no. 6: 585-594.

[3] Armstrong, R. D., D. S. Kung, P. Sinha, and A. A. Zoltners. 1983. A computational study of a multiple-choice knapsack algorithm. *ACM Transactions on Mathematical Software* 9, no. 2: 184-198.

[4] Averbakh, Igor and Oded Berman. 1997. Minimax regret p-center location on a network with demand uncertainty. *Location Science* 5, no. 4: 247-254.

[5] Averbakh, Igor and Oded Berman. 2000. Minmax regret median location on a network under uncertainty. *INFORMS Journal on Computing* 12, no. 2: 104-110.

[6] Balinski, M. L. 1965. Integer programming: Methods, uses, computation. *Management Science* 12, no. 3: 253-313.

[7] Ball, Michael O. 1979. Computing network reliability. *Operations Research* 27, no. 4: 823-838.

[8] Ball, Michael O. and Feng L. Lin. 1993. A reliability model applied to emergency service vehicle location. *Operations Research* 41, no. 1: 18-36.

[9] Barahona, Francisco and David Jensen. 1998. Plant location with minimum inventory. *Mathematical Programming* 83: 101-111.

[10] Barahona, Francisco and Fabian Chudak. 1999a. *Near-optimal solutions to large scale facility location problems*. Yorktown Heights, NY: IBM Research Division, T.J. Watson Research Center. IBM Research Report.

[11] Barahona, Francisco and Fabian Chudak. 1999b. *Solving large scale uncapacitated location problems*. Yorktown Heights, NY: IBM Research Division, T.J. Watson Research Center. IBM Research Report.

[12] Barahona, Francisco and Ranga Anbil. 2000. The volume algorithm: Producing primal solutions with a subgradient method. *Mathematical Programming Series A* 87: 385-399.

[13] Barcelo, Jaime, Elena Fernandez, and Kurt O. Jörnsten. 1991. Computational results from a new Lagrangean relaxation algorithm for the capacitated plant location problem. *European Journal of Operational Research* 53, no. 1: 38-45.

[14] Bean, James C., Julia L. Higle, and Robert L. Smith. 1992. Capacity expansion under uncertain demands. *Operations Research* 40, no. 2 supp.: S210-S216.

[15] Beasley, J. E. 1993. Lagrangean heuristics for location problems. *European Journal of Operational Research* 65, no. 3: 383-399.

[16] Berman, O. and B. LeBlanc. 1984. Location-relocation of mobile facilities on a stochastic network. *Transportation Science* 18, no. 4: 315-330.

[17] Berman, Oded, Richard C. Larson, and Samuel S. Chiu. 1985. Optimal server location on a network operating as an $M/G/1$ queue. *Operations Research* 33, no. 4: 746-771.

[18] Berman, Oded and Dimitri Krass. 2001. Facility location problems with stochastic demands and congestion. In *Facility location: Applications and theory*, ed. Zvi Drezner and H. W. Hamacher, 331-373. New York: Springer-Verlag.

[19] Bertsimas, Dimitris J., Patrick Jaillet, and Amedeo R. Odoni. 1990. A priori optimization. *Operations Research* 38, no. 6: 1019-1033.

[20] Bienstock, D., E. F. Brickell, and C. L. Monma. 1990. On the structure of minimum-weight k-connected spanning networks. *SIAM Journal on Discrete Mathematics* 3, no. 3: 320-329.

[21] Birge, John R. and François Louveaux. 1997. *Introduction to stochastic programming*. New York: Springer.

[22] Blanchini, Franco, Franca Rinaldi, and Walter Ukovich. 1997. A network design problem for a distribution system with uncertain demands. *SIAM Journal on Optimization* 7, no. 2: 560-578.

[23] Bramel, Julien and David Simchi-Levi. 1997. *The logic of logistics: Theory, algorithms, and applications for logistics management*. Springer series in operations research. New York: Springer.

[24] Burkard, Rainer E. and Helidon Dollani. 2001. Robust location problems with pos/neg weights on a tree. *Networks* 38, no. 2: 102-113.

[25] Carbone, Robert. 1974. Public facilities location under stochastic demand. *INFOR* 12, no. 3: 261-270.

[26] Carson, Yolanda M. and Rajan Batta. 1990. Locating an ambulance on the Amherst campus of the State University of New York at Buffalo. *Interfaces* 20, no. 5: 43-49.

[27] Chen, Bintong and Chin-Shien Lin. 1998. Minmax-regret robust 1-median location on a tree. *Networks* 31, no. 2: 93-103.

[28] Cheung, Raymond K.-M. and Warren B. Powell. 1996. Models and algorithms for distribution problems with uncertain demands. *Transportation Science* 30, no. 1: 43-59.

[29] Chopra, Sunil and Peter Meindl. 2001. *Supply chain management: Strategy, planning, and operation*. Upper Saddle River, NJ: Prentice Hall.

[30] Christofides, N. and J. E. Beasley. 1982. A tree-search algorithm for the p-median problem. *European Journal of Operational Research* 10, no. 2: 196-204.

[31] Christofides, N. and J. E. Beasley. 1983. Extensions to a Lagrangean relaxation approach for the capacitated warehouse location problem. *European Journal of Operational Research* 12, no. 1: 19-28.

[32] Church, Richard and Charles ReVelle. 1974. The maximal covering location problem. *Papers of the Regional Science Association* 32: 101-118.

[33] Cohon, Jared L. 1978. *Multiobjective programming and planning*. New York: Academic Press.

[34] Colbourn, C. J. 1987. *The combinatorics of network reliability*. The international series of monographs on computer science. New York: Oxford University Press.

[35] Cornuejols, Gerard, Marshall L. Fisher, and George L. Nemhauser. 1977. Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Science* 23, no. 8: 789-810.

[36] Cornuejols, G., R. Sridharan, and J.M. Thizy. 1991. A comparison of heuristics and relaxations for the capacitated plant location problem. *European Journal of Operational Research* 50: 280-297.

[37] Current, John, Samuel Ratick, and Charles ReVelle. 1997. Dynamic facility location when the total number of facilities is uncertain: A decision analysis approach. *European Journal of Operational Research* 110, no. 3: 597-609.

[38] Current, J., M. S. Daskin, and D. Schilling. 2001. Discrete network location models. In *Facility location: Applications and theory*, ed. Zvi Drezner and H. W. Hamacher, 83-120. New York: Springer-Verlag.

[39] Daniels, Richard L. and Panagiotis Kouvelis. 1995. Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science* 41, no. 2: 363-376.

[40] Darby-Dowman, Kenneth and Holly S. Lewis. 1988. Lagrangian relaxation and the single-source capacitated facility-location problem. *Journal of the Operational Research Society* 39, no. 11: 1035-1040.

[41] Darlington, J., C.C. Pantelides, B. Rustem, and B.A. Tanyi. 2000. Decreasing the sensitivity of open-loop optimal solutions in decision making under uncertainty. *European Journal of Operational Research* 121: 343-362.

[42] Daskin, Mark S. 1982. Application of an expected covering model to emergency medical service system design. *Decision Sciences* 13: 416-439.

[43] Daskin, Mark S. 1983. A maximum expected covering location model: Formulation, properties and heuristic solution. *Transportation Science* 17, no. 1: 48-70.

[44] Daskin, M. S., K. Hogan, and C. ReVelle. 1988. Integration of multiple, excess, backup, and expected covering models. *Environment and Planning B* 15, no. 1: 15-35.

[45] Daskin, Mark S., Wallace J. Hopp, and Benjamin Medina. 1992. Forecast horizons and dynamic facility location planning. *Annals of Operations Research* 40: 125-151.

[46] Daskin, Mark S. 1995. *Network and discrete location: Models, algorithms, and applications*. New York: Wiley.

[47] Daskin, Mark S., Susan M. Hesse, and Charles S. ReVelle. 1997. α-reliable P-minimax regret: A new model for strategic facility location modeling. *Location Science* 5, no. 4: 227-246.

[48] Daskin, Mark S. and Susan H. Owen. 1999. Location models in transportation. In *Handbook of transportation science*, ed. Randolph W. Hall, 311-360. Boston: Kluwer Academic.

[49] Daskin, Mark S., Collette R. Coullard, and Zuo-Jun Max Shen. 2002. An inventory-location model: Formulation, solution algorithm and computational results. *Annals of Operations Research* 110: 83-106.

[50] Davis, P.S. and T.L. Ray. 1969. A branch-bound algorithm for the capacitated facilities location problem. *Naval Research Logistics Quarterly* 16: 331-344.

[51] Drezner, Zvi. 1995. *Facility location: A survey of applications and methods.* Springer series in operations research. New York: Springer.

[52] Eiselt, H. A., M. Gendreau, and G. Laporte. 1996. Optimal location of facilities on a network with an unreliable node or link. *Information Processing Letters* 58, no. 2: 71-74.

[53] Eppen, Gary D. 1979. Effects of centralization on expected costs in a multi-location newsboy problem. *Management Science* 25, no. 5: 498-501.

[54] Eppen, Gary D., R. Kipp Martin, and Linus Schrage. 1989. A scenario approach to capacity planning. *Operations Research* 37, no. 4: 517-527.

[55] Erlebacher, Steven J. and Russell D. Meller. 2000. The interaction of location and inventory in designing distribution systems. *IIE Transactions* 32: 155-166.

[56] Erlenkotter, Donald. 1978. A dual-based procedure for uncapacitated facility location. *Operations Research* 26, no. 6: 992-1009.

[57] Fisher, Marshall L. 1981. The Lagrangian relaxation method for solving integer programming problems. *Management Science* 27, no. 1: 1-18.

[58] Fisher, Marshall L. 1985. An applications oriented guide to Lagrangian relaxation. *Interfaces* 15, no. 2: 10-21.

[59] Fortz, B. and M. Labbé. 2002. Polyhedral results for two-connected networks with bounded rings. *Mathematical Programming Series A* 93, no. 1: 27-54.

[60] Frank, H. 1966. Optimum locations on a graph with probabilistic demands. *Operations Research* 14, no. 3: 409-421.

[61] Frank, H. 1967. Optimum locations on graphs with correlated normal demands. *Operations Research* 15, no. 3: 552-557.

[62] França, P. M. and H. P. L. Luna. 1982. Solving stochastic transportation-location problems by generalized Benders decomposition. *Transportation Science* 16, no. 2: 113-126.

[63] Gendreau, Michel, Gilbert Laporte, and René Séguin. 1996. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research* 44, no. 3: 469-477.

[64] Geoffrion, A. M. and G. W. Graves. 1974. Multicommodity distribution system design by Benders decomposition. *Management Science* 20, no. 5: 822-844.

[65] Geoffrion, A.M. 1974. Lagrangean relaxation for integer programming. *Mathematical Programming Study* 2: 82-114.

[66] Geoffrion, A. and R. McBride. 1978. Lagrangean relaxation applied to capacitated facility location problems. *AIIE Transactions* 10, no. 1: 40-47.

[67] Glover, Fred. 1975. Surrogate constraint duality in mathematical programming. *Operations Research* 23, no. 3: 434-451.

[68] Glover, Fred. 1986. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research* 13: 533-549.

[69] Graves, S.C., A.H.G. Rinnooy Kan, and P.H. Zipkin. 1993. *Logistics of production and inventory.* Amsterdam: Elsevier Science Publishers.

[70] Grötschel, M., C. L. Monma, and M. Stoer. 1995. Polyhedral and computational investigations for designing communication networks with high survivability requirements. *Operations Research* 43, no. 6: 1012-1024.

[71] Guignard, Monique and Siwhan Kim. 1987. Lagrangean decomposition: A model yielding strong Lagrangean bounds. *Mathematical Programming* 39: 215-228.

[72] Gupta, Shiv K. and Jonathan Rosenhead. 1968. Robustness in sequential investment decisions. *Management Science* 15, no. 2: B18-B29.

[73] Gutiérrez, Genaro J. and Panagiotis Kouvelis. 1995. A robustness approach to international sourcing. *Annals of Operations Research* 59: 165-193.

[74] Gutiérrez, Genaro J., Panagiotis Kouvelis, and Abbas A. Kurawarwala. 1996. A robustness approach to uncapacitated network design problems. *European Journal of Operational Research* 94: 362-376.

[75] Haight, Robert G., Katherine Ralls, and Anthony M. Starfield. 2000. Designing species translocation strategies when population growth and future funding are uncertain. *Conservation Biology* 14, no. 5: 1298-1307.

[76] Hakimi, S.L. 1964. Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research* 12: 450-459.

[77] Hakimi, S.L. 1965. Optimum distribution of switching centers in a communication network and some related graph theoretic problems. *Operations Research* 13: 462-475.

[78] Hanink, Dean M. 1984. A portfolio theoretic approach to multiplant location analysis. *Geographical Analysis* 16, no. 2: 149-161.

[79] Hanjoul, Pierre and Dominique Peeters. 1985. A comparison of two dual-based procedures for solving the p-median problem. *European Journal of Operational Research* 20, no. 3: 387-396.

[80] Hobbs, Benjamin F., Michael H. Rothkopf, Richard P. O'Neill, and Hung-po Chao, eds. 2001. *The next generation of electric power unit commitment models*. International series in operations research and management science. Boston: Kluwer Academic Publishers.

[81] Hodder, James E. 1984. Financial market approaches to facility location under uncertainty. *Operations Research* 32, no. 6: 1374-1380.

[82] Hodder, James E. and James V. Jucker. 1985. A simple plant-location model for quantity-setting firms subject to price uncertainty. *European Journal of Operational Research* 21: 39-46.

[83] Holmberg, Kaj. 1998. Creative modeling: Variable and constraint duplication in primal-dual decomposition methods. *Annals of Operations Research* 82: 355-390.

[84] Hooker, J.N. and R.S. Garfinkel. 1989. On the vector assignment p-median problem. *Transportation Science* 23, no. 2: 139-140.

[85] Huchzermeier, Arnd and Morris A. Cohen. 1996. Valuing operational flexibility under exchange rate risk. *Operations Research* 44, no. 1: 100-113.

[86] Hurter, Arthur P. and Joseph Stanislaus Martinich. 1989. *Facility location and the theory of production*. Boston: Kluwer Academic Publishers.

[87] Jaillet, Patrick. 1988. A priori solution of a traveling salesman problem in which a random subset of the customers are visited. *Operations Research* 36, no. 6: 929-936.

[88] Jaillet, Patrick. 1992. Shortest path problems with node failures. *Networks* 22: 589-605.

[89] Jornsten, Kurt and Mette Bjorndal. 1994. Dynamic location under uncertainty. *Studies in Regional and Urban Planning* 3: 163-184.

[90] Jucker, James V. and Robert C. Carlson. 1976. The simple plant-location problem under uncertainty. *Operations Research* 24, no. 6: 1045-1055.

[91] Killmer, K.A., G. Anandalingam, and S.A. Malcolm. 2001. Siting noxious facilities under uncertainty. *European Journal of Operational Research* 113: 596-607.

[92] Klincewicz, John G. and Hanan Luss. 1986. A Lagrangian relaxation heuristic for capacitated facility location with single-source constraints. *Journal of the Operational Research Society* 37, no. 5: 495-500.

[93] Kogut, Bruce and Nalin Kulatilaka. 1994. Operating flexibility, global manufacturing, and the option value of a multinational network. *Management Science* 40, no. 1: 123-139.

[94] Kouvelis, Panagiotis, Abbas A. Kurawarwala, and Genaro J. Gutiérrez. 1992. Algorithms for robust single and multiple period layout planning for manufacturing systems. *European Journal of Operational Research* 63: 287-303.

[95] Kouvelis, Panagiotis and Gang Yu. 1997. *Robust discrete optimization and its applications*. Boston: Kluwer Academic Publishers.

[96] Laguna, Manuel, Pilar Lino, Angeles Pérez, Sacramento Quintanilla, and Vicente Valls. 2000. Minimizing weighted tardiness of jobs with stochastic interruptions in parallel machines. *European Journal of Operational Research* 127: 444-457.

[97] Laporte, Gilbert, François V. Louveaux, and Luc van Hamme. 1994. Exact solution to a location problem with stochastic demands. *Transportation Science* 28, no. 2: 95-103.

[98] Larson, Richard C. 1974. A hypercube queuing model for facility location and redistricting in urban emergency services. *Computers and Operations Research* 1: 67-95.

[99] Larson, Richard C. 1975. Approximating the performance of urban emergency service systems. *Operations Research* 23, no. 5: 845-868.

[100] Louveaux, François and Jacques-François Thisse. 1985. Production and location of a network under demand uncertainty. *Operations Research Letters* 4, no. 4: 145-149.

[101] Louveaux, F. V. 1986. Discrete stochastic location models. *Annals of Operations Research* 6: 23-34.

[102] Louveaux, François V. and D. Peeters. 1992. A dual-based procedure for stochastic facility location. *Operations Research* 40, no. 3: 564-573.

[103] Lowe, Timothy J., Richard E. Wendell, and Gang Hu. 1999. Screening location strategies to reduce exchange rate risk. Preprint.

[104] Lynn, Barry. 2002. Unmade in America: The true cost of a global assembly line. *Harper's*, 33-41.

[105] Manne, Alan S. 1961. Capacity expansion and probabilistic growth. *Econometrica* 29, no. 4: 632-649.

[106] Marín, Alfredo and Blas Pelegrín. 1999. Applying Lagrangian relaxation to the resolution of two-stage location problems. *Annals of Operations Research* 86: 179-198.

[107] Mausser, Helmut E. and Manuel Laguna. 1999a. A heuristic to minimax absolute regret for linear programs with interval objective function coefficients. *European Journal of Operational Research* 117: 157-174.

[108] Mausser, Helmut E. and Manuel Laguna. 1999b. Minimising the maximum relative regret for linear programmes with interval objective function coefficients. *Journal of the Operational Research Society* 50, no. 10: 1063-1070.

[109] Mirchandani, Pitu B. and Amedeo R. Odoni. 1979. Locations of medians on stochastic networks. *Transportation Science* 13, no. 2: 85-97.

[110] Mirchandani, Pitu B. 1980. Locational decisions on stochastic networks. *Geographical Analysis* 12, no. 2: 172-183.

[111] Mirchandani, Pitu B., Aissa Oudjit, and Richard T. Wong. 1985. 'Multidimensional' extensions and a nested dual approach for the m-median problem. *European Journal of Operational Research* 21, no. 1: 121-137.

[112] Monma, Clyde L. and David F. Shallcross. 1989. Methods for designing communications networks with certain two-connected survivability constraints. *Operations Research* 37, no. 4: 531-541.

[113] Monma, Clyde L., Beth Spellman Munson, and William R. Pulleyblank. 1990. Minimum-weight two-connected spanning networks. *Mathematical Programming* 46, no. 2: 153-171.

[114] Mulvey, John M., Robert J. Vanderbei, and Stavros A. Zenios. 1995. Robust optimization of large-scale systems. *Operations Research* 43, no. 2: 264-281.

[115] Nahmias, Steven. 2001. *Production and operations analysis*. Boston: McGraw-Hill Irwin.

[116] Nakagawa, Y., M. Kitao, M. Tsuji, and Y. Teraoka. 2001. Calculating the upper bound of the multiple-choice knapsack problem. *Electronics and Communications in Japan Part 3* 84, no. 7: 22-27.

[117] Nauss, Robert M. 1978a. The 0-1 knapsack problem with multiple choice constraints. *European Journal of Operational Research* 2: 125-131.

[118] Nauss, Robert M. 1978b. An improved algorithm for the capacitated facility location problem. *Journal of the Operational Research Society* 29, no. 12: 1195-1201.

[119] Nemhauser, George L. and Laurence A. Wolsey. 1988. *Integer and combinatorial optimization*. New York: Wiley.

[120] Nozick, Linda K. and Mark A. Turnquist. 1998. Integrating inventory impacts into a fixed-charge model for locating distribution centers. *Transportation Research Part E* 34, no. 3: 173-186.

[121] Nozick, Linda K. and Mark A. Turnquist. 2001. Inventory, transportation, service quality and the location of distribution centers. *European Journal of Operational Research* 129: 362-371.

[122] Nozick, Linda K. and Mark A. Turnquist. 2001. A two-echelon inventory allocation and distribution center location analysis. *Transportation Research Part E* 37: 421-441.

[123] Nozick, L.K. 2001. The fixed charge facility location problem with coverage restrictions. *Transportation Research Part E* 37: 281-296.

[124] Owen, Susan Hesse and Mark S. Daskin. 1998. Strategic facility location: A review. *European Journal of Operational Research* 111, no. 3: 423-447.

[125] Owen, Susan Hesse. 1999. Scenario planning approaches to facility location: Models and solution methods. Ph.D. diss., Northwestern University.

[126] Paraskevopoulos, Dimitris, Elias Karakitsos, and Berc Rustem. 1991. Robust capacity planning under uncertainty. *Management Science* 37, no. 7: 787-800.

[127] Pisinger, David. 1995. A minimal algorithm for the multiple-choice knapsack problem. *European Journal of Operational Research* 83, no. 2: 394-410.

[128] Raghavan, P and C.D. Thompson. 1987. Randomized rounding. *Combinatorica* 7: 365-374.

[129] ReVelle, C.S. and R.W. Swain. 1970. Central facilities location. *Geographical Analysis* 2: 30-42.

[130] ReVelle, Charles and Kathleen Hogan. 1989. The maximum availability location problem. *Transportation Science* 23, no. 3: 192-200.

[131] ReVelle, C. and J. C. Williams. 2001. Reserve design and facility siting. In *Facility location: Applications and theory*, ed. Zvi Drezner and H. W. Hamacher, 310-330. New York: Springer-Verlag.

[132] Rolland, Eric, David A. Schilling, and John R. Current. 1996. An efficient tabu search procedure for the p-median problem. *European Journal of Operational Research* 96: 329-342.

[133] Rosenblatt, Meir J. and Hau L. Lee. 1987. A robustness approach to facilities design. *International Journal of Production Research* 25, no. 4: 479-486.

[134] Rosenhead, Jonathan, Martin Elton, and Shiv K. Gupta. 1972. Robustness and optimality as criteria for strategic decisions. *Operational Research Quarterly* 23, no. 4: 413-431.

[135] Schilling, David A. 1982. Strategic facility planning: The analysis of options. *Decision Sciences* 13: 1-14.

[136] Schrage, Linus. 1975. Implicit representation of variable upper bounds in linear programming. *Mathematical Programming Study* 4: 118-132.

[137] Serra, D., S. Ratick, and C. ReVelle. 1996. The maximum capture problem with uncertainty. *Environment and Planning B* 23: 49-59.

[138] Serra, Daniel and Vladimir Marianov. 1998. The p-median problem in a changing network: The case of Barcelona. *Location Science* 6: 383-394.

[139] Sheffi, Yossi. 2001. Supply chain management under the threat of international terrorism. *International Journal of Logistics Management* 12, no. 2: 1-11.

[140] Shen, Zuo-Jun Max. 2000. Efficient algorithms for various supply chain problems. Ph.D. diss., Northwestern University.

[141] Shen, Zuo-Jun Max, Collette R. Coullard, and Mark S. Daskin. 2003. A joint location-inventory model. *Transportation Science* 37, no. 1: 40-55.

[142] Sheppard, E. S. 1974. A conceptual framework for dynamic location-allocation analysis. *Environment and Planning A* 6: 547-564.

[143] Shier, Douglas R. 1991. *Network reliability and algebraic structures.* Oxford: Clarendon Press.

[144] Shooman, Martin L. 2002. *Reliability of computer systems and networks: Fault tolerance, analysis, and design.* New York: John Wiley and Sons.

[145] Shu, Jia, Chung-Piaw Teo, and Zuo-Jun Max Shen. 2001. Stochastic transportation-inventory network design problem. Preprint.

[146] Simchi-Levi, D., L.V. Snyder, and M. Watson. 2002. Strategies for uncertain times. *Supply Chain Management Review* 6, no. 1: 11-12.

[147] Sinha, Prabhakant and Andris A. Zoltners. 1979. The multiple-choice knapsack problem. *Operations Research* 27, no. 3: 503-515.

[148] Sridharan, R. 1991. A Lagrangian heuristic for the capacitated plant location problem with side constraints. *Journal of the Operational Research Society* 42, no. 7: 579-585.

[149] Swamy, Chaitanya and David B. Shmoys. 2003. Fault-tolerant facility location. *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*: 735-736.

[150] Teitz, Michael B. and Polly Bart. 1968. Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations Research* 16, no. 5: 955-961.

[151] Teo, Chung-Piaw, Jihong Ou, and Mark Goh. 2001. Impact on inventory costs with consolidation of distribution centers. *IIE Transactions* 33, no. 2: 99-110.

[152] Trafalis, Theodore B., Tsutomu Mishina, and Bobbie L. Foote. 1999. An interior point multiobjective programming approach for production planning with uncertain information. *Computers and Industrial Engineering* 37: 631-648.

[153] Vairaktarakis, George L. and Panagiotis Kouvelis. 1999. Incorporation dynamic aspects and uncertainty in 1-median location problems. *Naval Research Logistics* 46, no. 2: 147-168.

[154] Van Roy, Tony J. 1983. Cross decomposition for mixed integer programming. *Mathematical Programming* 25: 46-63.

[155] Van Roy, Tony J. 1986. A cross decomposition algorithm for capacitated facility location. *Operations Research* 34, no. 1: 145-163.

[156] Verter, Vedat and M. Cemal Dincer. 1992. An integrated evaluation of facility location, capacity acquisition, and technology selection for designing global manufacturing strategies. *European Journal of Operational Research* 60: 1-18.

[157] Vladimirou, Hercules and Stavros A. Zenios. 1997. Stochastic linear programs with restricted recourse. *European Journal of Operational Research* 101: 177-192.

[158] Weaver, Jerry R. and Richard L. Church. 1983. Computational procedures for location problems on stochastic networks. *Transportation Science* 17, no. 2: 168-180.

[159] Weaver, Jerry R. and Richard L. Church. 1985. A median location model with nonclosest facility service. *Transportation Science* 19, no. 1: 58-74.

[160] Yu, Gang. 1997. Robust economic order quantity models. *European Journal of Operational Research* 100: 482-493.

[161] Yu, Gang and Jian Yang. 1998. On the robust shortest path problem. *Computers and Operations Research* 25, no. 6: 457-468.

[162] Yu, Chian-Son and Han-Lin Li. 2000. A robust optimization model for stochastic logistics problems. *International Journal of Production Economics* 64: 385-397.

[163] Zipkin, P.H. 1997. *Foundations of inventory management*. Burr Ridge, Ill.: Irwin.