The Reliable Facility Location Problem: Formulations, Heuristics, and Approximation Algorithms

Zuo-Jun Max Shen

Department of Industrial Engineering and Operations Research,

University of California, Berkeley, CA

Roger Lezhou Zhan

Department of Industrial and Systems Engineering, University of Florida

Jiawei Zhang

IOMS-Operations Management, Stern School of Business, New York University

July 7, 2007

Abstract

We study the reliable facility location problem in which some facilities are subject to failure from time to time. If a facility fails, customers originally assigned to it have to be reassigned to other (operational) facilities. We formulate this problem as a two-stage stochastic program and then a nonlinear integer program. Several heuristics that can produce near-optimal solutions are proposed for this NP-hard problem. For a special case where the probability that a facility fails is a constant (independent of the facility), we provide an approximation algorithm with a worst-case bound of 2.674. The effectiveness of our heuristics is tested by extensive computational studies, which also lead to some managerial insights.

1 Introduction

Facility location models have been extensively studied in the literature. Different kinds of facilities have been modeled, such as routers or servers in a communication network, warehouses or distribution centers in a supply chain, hospitals or airports in a public service system. Facility location models typically try to determine where to locate the facilities among a set of candidate sites, and how to assign 'customers' to the facilities, so that the total cost can be minimized or the total profit can be maximized (e.g., [11], [4], and [16]). Most models in the literature have treated facilities as if they would never fail; in other words, they are completely reliable. We will relax this assumption in this paper.

One of the most studied location models is the so-called Uncapacitated Facility Location Problem (UFLP). In the UFLP, we are given a set of demand points, a set of candidate sites, the cost of opening a facility at each location, and the cost of connecting each demand point to any facility. The objective is to open a set of facilities from the candidate sites and assign each demand point to an open facility so as to minimize the total facility opening and connection costs.

The UFLP and its generalizations are NP-hard, i.e., unless $P = NP$ they do not admit polynomial-time algorithms to find an optimal solution. There is a vast literature on these NP-hard facility location problems and many solution approaches (integer programming, meta-heuristics, approximation algorithms, etc.) have been developed in the last four decades. One common assumption in this literature is that the input parameters of the problems (costs, demands, facility capacities, etc.) are deterministic. However, such assumptions may not be valid in many realistic situations since many input parameters in the model are uncertain during the decision-making process.

The uncertainties can be generally classified into three categories: provider-side uncertainty, receiver-side uncertainty, and in-between uncertainty. The provider-side uncertainty may capture the randomness in facility capacity and the reliability of facilities, etc.; the receiver-side uncertainty can be the randomness in demands; and the in-between uncertainty may be represented by the random travel time, transportation cost, etc. Most stochastic facility location models focus on the receiver-side and in-between uncertainties ([18]). The common feature of the receiver-side and in-between uncertainties is that the uncertainty does not change the topology of the provider-receiver network once the facilities have been built. However, this is not the case if the built facilities are subject to fail (provider-side uncertainty). If a facility fails, customers originally assigned to it have to be reassigned to other (operational) facilities, and thus the connection cost changes (usually increases).

We focus on the reliability issue of provider-side uncertainty in this paper. The uncertainty is modeled using two different approaches: 1) by a set of scenarios that specify which subset of the facilities will become non-operational; or 2) by an individual and independent failure probability inherent in each facility. Although each demand point needs to be served by one operational facility only, it should be assigned to a group of facilities that are ordered by levels: in the event of the lowest level facility becoming non-operational, the demand can then be served by the next level facility that is operational; and so on. If all operational facilities are too far away from a demand point, one may choose not to serve this demand point by paying a penalty cost. The objective is thus to minimize the facility opening cost plus the *expected* connection and penalty costs. This problem will be referred to as the Uncapacitated Reliable Facility Location Problem (URFLP).

The URFLP is clearly NP-hard as it generalizes the UFLP. We propose several heuristics to solve the URFLP. We also provide a 2.647-approximation algorithm for a special case where the probability of a facility failing is independent of the facility. Designing approximation algorithms for the UFLP and its variations has recently received considerable attentions from the research community. However, to the best of our knowledge, our paper presents the first approximation algorithm for stochastic facility location problems with provider-side uncertainty.

The rest of this paper is organized as follows. In Section 2, we review the related literature and provide some basic background for our model. In Section 3, we present two formulations for the model and discuss their properties. An approximation algorithm for a special case of the model is presented in Section 4. In Section 5, we discuss and evaluate different heuristics. In Section 6, we conclude the paper by suggesting several future research directions.

2 Literature Review

The importance of uncertainty in decision making has promoted a number of researchers to address stochastic facility location models (e.g., [18, 12]). However, as we pointed out in the Introduction, a majority of the current literature mainly deals with the receiver-side and/or in-between uncertainties. This includes [23], [6], [5], and [13] among others.

[19] is closely related to our paper. In [19], the authors assume that some facilities are perfectly reliable while others are subject to failure with the same probability. They formulate their problem as a linear integer program and propose a Lagrangian relaxation solution method. Another related

model is proposed in [1], which is based on the p-median problem rather than the UFLP.

There is also a strand of literature addressing the fortification of reliability for existing facilities (e.g., [14], [15], and [20]). These models typically use the interdiction-fortification p-median problem framework and are generally formulated as bilevel programming models. Their main focuses are to identify the *existing* critical facilities to protect under the events of disruption.

Our paper is also related to the literature on approximation algorithms for facility location problems (e.g. [17, 7, 8, 10, 3]). However, until very recently, these approximation algorithms mainly deal with deterministic problems. Approximation algorithms for the UFLP with stochastic demand have recently been proposed; see the survey by Shmoys and Swamy [21]. Another related paper [2] proposes an approximation algorithm for a facility location problem with stochastic demand and inventory. Our approximation algorithm makes use of the ideas from several papers [17, 7, 8, 10]. In particular, our paper is closely related to [7], which presents a 2.41-approximation algorithm for the so-called fault-tolerant facility location problem (FTFLP): every demand point must be served by several facilities where the number is specified, and a weighted linear combination is used to compute the connection costs. The FTFLP has been motivated by the reliability issue considered in our paper, but the failure probabilities of facilities are not explicitly modeled and no penalty cost is considered.

In our model, the failure probabilities are facility-specific, which significantly complicates the problem when formulating it as a mathematical program. We propose two different models: a scenario-based stochastic programming (SP) model and a nonlinear integer programming (NIP) model. The scenario-based model is attractive due to its structural simplicity and its ability to model dependence among random parameters. But the model becomes computationally expensive as the number of scenarios increases. If the number of scenarios is too large, the nonlinear integer programming based approach provides an alternative way to tackle the problem. We also consider the special case when the failure probabilities are not facility-specific. Notice that even under this assumption, our model still differs from that in [19] as we consider penalty cost. Furthermore, we also propose and analyze an approximation algorithm with constant worst-case bound guarantee for the model.

3 Formulations

We first introduce some common notation that will be used throughout the paper. Let D denote the set of clients or demand points and F denote the set of facilities. $|F|$ is the number of the facilities. Let f_i be the facility cost to open facility i, d_j be the demand of client j, and c_{ij} be the service cost if j is serviced by facility i. The service costs, c_{ij}, are assumed to form a metric, i.e., they satisfy triangle inequalities. For each client $j \in D$, if it is not served by any open and operational facility, then a penalty cost r_j will be incurred.

3.1 Scenario Based Model

We first discuss a scenario based approach to model the URFLP. Given a finite set of scenarios, where each scenario specifies the set of operational facilities, we can formulate the URFLP as a two-stage stochastic program with recourse. The first stage decision is to determine which facilities to open before knowing which facilities will be operational. When the uncertainty is resolved, the clients (demand points) will be assigned to the operational facilities, which is the second stage (recourse) decision. In this model, we are not allowed to build new facilities in the second stage. In other words, no remedy can be made to the first stage decision, except optimally assigning the clients to the operational facilities. The objective is to minimize the total expected cost which includes the first stage cost and the expected second stage cost. The expectation is taken over all scenarios according to a specific distribution.

Let $\mathcal{S}$ be the set of scenarios. For any $A \in \mathcal{S}$, let p_A be the probability that scenario A happens. Then the URFLP can be formulated as the following two-stage stochastic program.

$$\text{minimize} \quad \sum_{i \in F} f_i y_i + \sum_{A \in \mathcal{S}} p_A g_A(y) \text{ subject to } y_i \in \{0,1\}, \tag{1}$$

$$\text{where} \quad g_A(y) = \min \sum_{j \in D} \sum_{i \in F} d_j c_{ij} x_{ij}^A + \sum_{j \in D} d_j r_j z_j^A \tag{2}$$

$$\text{s.t.} \sum_{i \in F} x_{ij}^A + z_j^A = 1, \quad \forall j \in D \tag{3}$$

$$x_{ij}^A \le y_i, \quad \forall i \in F, j \in D \tag{4}$$

$$x_{ij}^A \le I_{A,i}, \quad \forall i \in F, j \in D \tag{5}$$

$$x_{ij}^A, z_j^A \in \{0,1\}. \tag{6}$$

In the above formulation, the binary variable y_i indicates if facility i is opened in the first stage. Parameter $I_{A,i}$ indicates if facility i is operational under scenario A, which is an input regardless of the value of y_i. Variable x_{ij}^A is the assignment variable which indicates whether client j is assigned to facility i in scenario A or not. Finally, the variable z_j^A indicates whether client j receives service at all or is subject to a penalty. The objective in the formulation, i.e. 1, is to minimize the sum of the fixed cost and the expected second stage cost. The objective of the second stage, i.e. function 2 is to minimize the service and penalty cost. Constraints (3) ensure that client j is either assigned to a facility or subject to a penalty at each level k in Scenario A. Constraints (4) and (5) make sure that no client is assigned to an unopen facility or a nonfunctional facility respectively.

It is straightforward to show that the formulation (1) is equivalent to the following mathematical program.

(*URFLP-SP*)

$$\begin{aligned}
\text{minimize} \quad & \sum_{i\in F} f_i y_i + \sum_{A\subseteq \mathcal{S}} p_A \left(\sum_{j\in D}\sum_{i\in F} d_j c_{ij} x_{ij}^A + \sum_{j\in D} d_j r_j z_j^A \right) \\
\text{s.t.} \quad & \sum_{i\in F} x_{ij}^A + z_j^A = 1, \quad \forall j\in D, A\subseteq \mathcal{S} \\
& x_{ij}^A \le y_i I_{A,i}, \quad \forall i\in F, j\in D, A\subseteq \mathcal{S} \\
& y_i, x_{ij}^A, z_j^A \in \{0,1\}.
\end{aligned}$$

One advantage of the scenario based formulation is that it can easily capture the dependence of the failure probabilities of different facilities by properly defining the scenarios. If the number of scenarios is not too large, it is possible to solve the URFLP-SP reasonably efficiently and effectively.

However, when the failure probabilities are independent, the possible number of scenarios can be extremely large. Therefore, the number of variables and constraints in the URFLP-SP is exponentially large accordingly, which makes it extremely difficult to solve. Under this situation, we propose an alternative nonlinear integer programming formulation and an efficient solution algorithm. We discuss this alternative formulation in the next subsection.

3.2 Nonlinear Integer Programming Model

In this subsection, we assume that the failure probabilities of the facilities are independent. Let p_i denote the probability that facility i fails. Without loss of generality, we assume that $0 \leq p_i < 1$. The major difficulty here is to compute the expected service cost for each client. In order to overcome this difficulty, we extend Snyder and Daskin's formulation [19] to a more general setting. Comparing to [19], the URFLP can be interpreted slightly differently as follows. Each client should be assigned to a set of facilities initially. The facilities assigned to any client can be differentiated by the levels: in case a lower level facility fails, the next level facility, if operational, will provide service instead.

Mathematically, we define two types of new binary variables x_{ij}^k, z_j^k to capture different level of facilities for a client j. In particular, $x_{ij}^k = 1$ if facility i is the k-th level backup facility of client j and otherwise, $x_{ij}^k = 0$; $z_j^k = 1$ if j has $(k-1)$-th backup facility, but has no k-th backup facility so that j incurs a penalty cost at level k.

Given the variables x_{ij}^k, z_j^k, one can compute the expected total service cost. Consider a client j and its expected service cost at its level-k facility. Client j is served by its level-k facility only if all its lower level assigned facilities become non-operational. For any facility l, if it is on the lower level (i.e, less than k) for demand node j, then $\sum_{s=1}^{k-1} x_{lj}^s = 1$, otherwise $\sum_{s=1}^{k-1} x_{lj}^s = 0$. It follows that for client j, the probability that all its lower level facilities fail is $\prod_{l \in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s}$. If j is severed by facility i, as j's level-k backup facility, then facility i has to be operational which occurs with probability $(1 - p_i)$. Therefore, the expected service cost of client j at level k is $\sum_{i \in F} d_j c_{ij} x_{ij}^k (1 - p_i) \prod_{l \in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s}$. Similarly, we can analyze the penalty cost of client j at level k, which is $\prod_{l \in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s} d_j r_j z_j^k$.

The above discussion leads to a non-linear integer programming formulation for the URFLP as follows.

(*URFLP-NIP*)

$$\text{minimize} \quad \sum_{i\in F} f_i y_i + \sum_{j\in D}\sum_{k=1}^{|F|}\sum_{i\in F} d_j c_{ij} x_{ij}^k (1-p_i) \prod_{l\in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s}$$
$$+\sum_{j\in D}\sum_{k=1}^{|F|+1}\prod_{l\in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s} d_j r_j z_j^k \tag{7}$$
$$\text{subject to} \quad \sum_{i\in F} x_{ij}^k + \sum_{t=1}^{k} z_j^t = 1, \quad \forall j \in D, k = 1, ..., |F|+1 \tag{8}$$
$$x_{ij}^k \le y_i, \quad \forall i \in F, j \in D, k = 1, ..., |F| \tag{9}$$
$$\sum_{k=1}^{|F|} x_{ij}^k \le 1, \quad \forall i \in F, j \in D \tag{10}$$
$$x_{ij}^k, z_j^k, y_i \in \{0,1\}. \tag{11}$$

The decision variables x_{ij}^k, z_j^k are defined earlier. The indicator variable $y_i = 1$ if facility i is open in the first stage; otherwise $y_i = 0$. The objective function (7) is the summation of the facility cost, the expected service cost, and the expected penalty cost. Constraints (8) ensure that client j is either assigned to a facility or subject to a penalty at each level k. Constraints (9) make sure that no client is assigned to an unopen facility. Constraints (10) prohibit a client from being assigned to a specific facility at more than one level. Note that Constraints (9) and Constraints (10) can be tightened as

$$\sum_{k=1}^{|F|} x_{ij}^k \le y_i \quad \forall i \in F, j \in D. \tag{12}$$

3.3 Model Properties

In formulation (URFLP-NIP), we do not explicitly require that a closer open facility be assigned as a lower level facility to a particular demand point. However, according to the following proposition, it is true that the level assignments among the open facilities are based on the relative distances between the demand point and the facilities regardless of the failure probabilities.

Proposition 1. *In any optimal solution to the (URFLP-NIP), for any client j, if $x_{uj}^k = x_{vj}^{k+1} = 1$, then $c_{uj} \le c_{vj}$.*

Proof. We prove the proposition by contradiction. Suppose $c_{uj} > c_{vj}$, we will show that by "swapping" the assignment of u and v, the objective function will strictly decrease.

In particular, if we set $x_{uj}^{k+1} = 1$ and $x_{vj}^{k} = 1$ with the values of other variables unchanged, we can compute the new objective value. It can be shown that the difference between the new objective value and the original one is

$$\begin{aligned} & d_j \left(c_{vj}(1-p_v)\bar{p_k} + c_{uj}(1-p_u)\bar{p_k}p_v\right) - d_j \left(c_{uj}(1-p_u)\bar{p_k} + c_{vj}(1-p_v)\bar{p_k}p_u\right) \\ = \quad & d_j(c_{vj} - c_{uj})(1-p_v)(1-p_u)\bar{p_k} \\ < 0, & \end{aligned}$$

where

$$\bar{p_k} = \prod_{l \in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s},$$

and the last inequality holds because $\bar{p_k} > 0$ and it is assumed that $p_u < 1$, $p_v < 1$, and $c_{vj} < c_{uj}$. This is clearly a contradiction to the optimality of the original solution. Therefore, $c_{uj} \leq c_{vj}$. $\square$

An implication of Proposition 1 is that if the set of open facilities is determined, then it is trivial to solve the level assignment problem for each client: assigning levels according to the relative distances of different facilities to the client. If at some level the distance is beyond the penalty cost, then no facility will be assigned at this level (and higher ones) and the demand node simply takes the (cheaper) penalty.

We would like to point out the relationship between formulation (URFLP-SP) and formulation (URFLP-NIP). Since these two formulations are just two ways of modeling the same problem, they should have the same minimum cost as long as the inputs to the two models are consistent. In formulation (URFLP-NIP), each facility i has independent failure probability p_i. This implies that there are $2^{|F|}$ scenarios and the probability that each of the scenarios will occur can be calculated. The corresponding values serve as inputs to formulation (URFLP-SP). We notice that, in formulation (URFLP-SP), it is straightforward to obtain an optimal second stage solution for a given first stage solution. In particular, at the second stage, every client will be assigned to and be served by an open and operational facility that is closest to the client at the lowest possible level; if the service cost is higher than the penalty cost, then it takes the penalty.

3.4 Uniform Failure Probabilities

Now we consider a special case of the URFLP where all facilities have the same failure probability, i.e., $p_i = p, \forall i \in F$. This assumption simplifies formulation (URFLP-NIP) considerably based on the following observation. Because $p_i = p \forall i \in F$, it is straightforward that $\prod_{l \in F} p_l^{\sum_{s=1}^{k-1} x_{lj}^s} = p^{k-1}$, which is independent of the values of x_{lj}^s. This property is implicitly used in a multi-objective formulation proposed in [19].

Based on the above observation, we are able to reduce formulation (URFLP-NIP) to a linear integer program as follows.

(*URFLP-IP*)

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in F} f_i y_i + \sum_{j \in D} \sum_{k=1}^{|F|} \sum_{i \in F} d_j c_{ij} x_{ij}^k (1-p) p^{k-1} + \sum_{j \in D} \sum_{k=1}^{|F|+1} p^{k-1} d_j r_j z_j^k & (13) \\ \text{subject to} \quad & \sum_{i \in F} x_{ij}^k + \sum_{t=1}^{k} z_j^t = 1, \quad \forall j \in D, k = 1, ..., |F|+1 \\ & \sum_{k=1}^{|F|} x_{ij}^k \leq y_i, \quad \forall i \in F, j \in D \\ & x_{ij}^k, z_j^k, y_i \in \{0, 1\}. \end{aligned}$$

In the next section, we shall present an approximation algorithm for (*URFLP-IP*). We find that it is more convenient to deal with a slightly different formulation of (*URFLP-IP*). In the new formulation, we introduce a new set of decision variables θ_j^k to replace z_j^k. Define $\theta_j^k = 1$ if j is not assigned to any facility as its k^{th} back-up facility. Also, for simplicity, we let $\theta_j^k = 1$ for all $k \geq |F|+1$. Note that θ_j^k are not decision variables when $k > |F|$.

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in F} f_i y_i + \sum_{j \in D} \sum_{k=1}^{|F|} \sum_{i \in F} d_j c_{ij} x_{ij}^k p^{k-1} (1-p) + \sum_{j \in D} \sum_{k=1}^{\infty} d_j r_j \theta_j^k p^{k-1} (1-p) & (14) \\ \text{subject to} \quad & \sum_{i \in F} x_{ij}^k + \theta_j^k = 1, \quad \forall j \in F, k = 1, ..., |F| \\ & \sum_{k=1}^{|F|} x_{ij}^k \leq y_i, \quad \forall i \in F, j \in D \\ & x_{ij}^k, \theta_j^k, y_i \in \{0, 1\}. \end{aligned}$$

We prove that the above integer program is equivalent to the formulation (URFLP-IP) as stated in Theorem 1. We refer the reader to Appendix A.1 for a proof.

Theorem 1. *Formulation (13) and formulation (14) are equivalent.*

4 Approximation Algorithms: Uniform Probabilities

In this section, we aim to propose a 2.674-approximation algorithm for the special case where the failure probabilities are uniform. We call it a (R_f, R_c, R_p)-approximation algorithm, if the algorithm produces a solution with cost no more than

$$R_f F^* + R_c C^* + R_p P^*,$$

where F^*, C^*, and P^* are the optimal facility, transportation, and penalty cost, respectively.

We take advantage of several results for the fault-tolerant version of the UFLP, where every demand point j must be served by k_j distinct facilities, a concept close to our level assignment. In [7], Guha *et al.* propose a couple of approximation algorithms for the fault-tolerant facility location problem using various rounding and greedy local-search techniques. One of the key results from Guha *et al.* [7] is summarized below.

Lemma 1. *For any vector (x, y) satisfying the following inequalities (the dimension of (x, y) should be clear from the inequalities)*

$$\sum_{i \in F} x_{ij}^k \geq 1, \quad \forall j \in D, k \leq k_j$$

$$\sum_{k=1}^{k_j} x_{ij}^k \leq y_i, \quad \forall i \in F, j \in D$$

$$x_{ij}^k \geq 0,$$

$$0 \leq y_i \leq 1,$$

and any $\alpha \in (0, 1]$, one can find an integer solution $(\tilde{x}, \tilde{y})$ satisfying the above inequalities so that

$$\sum_{i \in F} f_i \tilde{y}_i + \sum_{j \in D} \sum_{k=1}^{k_j} \sum_{i \in F} w_j^k d_j c_{ij} \tilde{x}_{ij}^k \leq \frac{\ln \frac{1}{\alpha}}{1 - \alpha} f_i y_i + \frac{3}{1 - \alpha} \sum_{j \in D} \sum_{k=1}^{k_j} \sum_{i \in F} w_j^k d_j c_{ij} x_{ij}^k.$$

In Lemma 1, w_j^k is the weighted factor at level k for demand node j. α is a parameter that we can choose to control the quality of approximation. Indeed, the approximation ratios of the algorithms of Guha *et al.* are functions of α. One can then choose the best α to minimize the approximation ratio.

We are now ready to present our algorithm for the special case of the URFLP. We first solve a linear programming relaxation of formulation (14).

$$\begin{aligned}
\text{minimize} \quad & \sum_{i\in F} f_i y_i + \sum_{j\in D}\sum_{k=1}^{|F|}\sum_{i\in F} d_j c_{ij} x_{ij}^k p^{k-1}(1-p) + \sum_{j\in D}\sum_{k=1}^{\infty} d_j r_j \theta_j^k p^{k-1}(1-p) && (15)\\
\text{subject to} \quad & \sum_{i\in F} x_{ij}^k + \theta_j^k = 1, \quad \forall j\in D, k=1,...,|F| && (16)\\
& \sum_{k=1}^{|F|} x_{ij}^k \le y_i, \quad \forall i\in F, j\in D && (17)\\
& x_{ij}^k \ge 0, \theta_j^k \ge 0. && (18)
\end{aligned}$$

Assume that (x, y, θ) is an optimal solution to this linear program. Our algorithm rounds the fractional solution (x, y, θ) to an integer solution $(\bar{x}, \bar{y}, \bar{\theta})$ that is feasible to formulation (14).

The algorithm is based on a property of the optimal fractional solution (x, y, θ), which is formalized in the following lemma. This lemma enables us to utilize known algorithms and analysis for the fault-tolerant facility location problem.

Lemma 2. *For each $j \in D$, the following two statements are true.*

(i). $\theta_j^k \le \theta_j^{k'}$ for any $1 \le k \le k'$.

(ii). If there exists k such that $0 < \theta_j^k < 1$, then $\theta_j^{k'} = 0$ and $\theta_j^{k''} = 1$ for $k' < k$ and $k'' > k$.

Proof. The proof is intuitive and similar to the proof of theorem 1, which is thus omitted here. □

We present our rounding procedure next. For each $j \in D$, assume k_j is the smallest integer such that $\theta_j^{k_j} > 0$.

The rounding procedure is carried out in two phases. The first phase rounds the optimal fractional solution (x, y, θ) to another fractional solution $(\hat{x}, \hat{y}, \hat{\theta})$, which is feasible to a linear programming relaxation of an appropriately defined fault-tolerant facility location problem. In the second phase, we use an algorithm for the fault-tolerant facility location problem to round the fractional solution $(\hat{x}, \hat{y}, \hat{\theta})$ to an integer solution $(\bar{x}, \bar{y}, \bar{\theta})$, which is feasible to formulation (14).

Phase I: Decomposition

- For every $j \in D$ and $i \in F$, let $\hat{\theta}_j^k = \theta_j^k$ and $\hat{x}_{ij}^k = x_{ij}^k$ for all $k \geq 1$ except for $k = k_j$.
- Choose a parameter $\delta \in (0, 1]$ whose value will be fixed later.
- Randomly generate a variable β that is uniformly distributed in $[0, \delta]$. For each $j \in D$ and $i \in F$, if $\theta_j^{k_j} \geq \beta$, then set

  $$\hat{\theta}_j^{k_j} = 1, m_j = k_j - 1, \hat{x}_{ij}^{k_j} = 0,$$

  otherwise set

  $$\hat{\theta}_j^{k_j} = 0, m_j = k_j, \hat{x}_{ij}^{k_j} = \frac{x_{ij}^{k_j}}{1 - \theta_j^{k_j}} = 1.$$

- For each $i \in F$, set

  $$\hat{y}^i = \max_{j \in D} \sum_{k=1}^{m_j} \hat{x}_{ij}^k.$$

Phase II: Solving Fault-Tolerant Problem

- For each $j \in D$ and $i \in F$, let $\bar{\theta}_j^k = 1$ and $\bar{x}_{ij}^k = 0$ for all $k > m_j$, and let $\bar{\theta}_j^k = 0$ for all $k \leq m_j$, where m_j is defined in Phase I of the algorithm.
- Use the algorithm(s) in [7] with a parameter $\alpha \in (0, 1]$ to round the solution $(\hat{x}, \hat{y})$ to a feasible solution of a fault-tolerant facility location problem, where a set of facilities is open such that each client j is served by at least m_j distinct open facilities.
- For each $i \in F$, set $\bar{y}_i = 1$ if facility i is open, and set $\bar{y}_i = 0$ otherwise.
- For each $j \in D$, if i is the k^{th} closest open facility to client j, then let $\bar{x}_{ij}^k = 1$, where $k \leq m_j$.
- Output the solution $(\bar{x}, \bar{y}, \bar{\theta})$.

This two-phase algorithm shall be referred to as Algorithm TP. It is obvious that the solution $(\bar{x}, \bar{y}, \bar{\theta})$ is feasible to formulation (14). We now establish a worst case approximation bound of our (randomized) algorithm, i.e., we shall show that the (expected) total cost is no more than a constant factor times the optimal cost. We bound the total penalty cost in Lemma 3, and bound the total facility and transportation costs in Lemma 4.

Lemma 3. *In Algorithm TP, the expected total penalty cost is bounded from above by*

$$\sum_{j\in D}\left(\frac{1}{\delta}d_j r_j p^{k_j-1}(1-p)\theta_j^{k_j}+\sum_{k=k_j+1}^{\infty}d_j r_j p^{k-1}(1-p)\theta_j^{k}\right).$$

Lemma 4. *For any $\alpha\in(0,1)$, the expected facility cost plus the expected transportation cost is no more than*

$$\frac{\ln\frac{1}{1-\delta}}{\delta}\cdot\left(\frac{\ln\frac{1}{\alpha}}{1-\alpha}F_{LP}+\frac{3}{1-\alpha}C_{LP}\right),$$

where F_{LP} and C_{LP} are the total facility cost and the total transportation cost, respectively, corresponding to the solution (x,y,θ) in the linear programming relaxation of formulation (14).

We refer the reader to Appendix A.2 for proofs of both lemmas. An immediate consequence of these two lemmas is the following corollary.

Corollary 1. *For any $\alpha\in(0,1)$ and $\delta\in(0,1)$, there is a $\left(\frac{\ln\frac{1}{1-\delta}}{\delta}\frac{\ln\frac{1}{\alpha}}{1-\alpha},\frac{\ln\frac{1}{1-\delta}}{\delta}\frac{3}{1-\alpha},\frac{1}{\delta}\right)$-approximation algorithm for the problem with uniform probabilities.*

By choosing $\alpha=0.049787057$, $\delta=0.271478197$, we derive that $\max\left(\frac{\ln\frac{1}{1-\delta}}{\delta}\frac{\ln\frac{1}{\alpha}}{1-\alpha},\frac{\ln\frac{1}{1-\delta}}{\delta}\frac{3}{1-\alpha},\frac{1}{\delta}\right)<$ 3.6836, which leads to the main result of this section.

Theorem 2. *The uniform case of URFLP admits a* 3.6836*-approximation algorithm.*

We use a technique called greedy improvement procedure (see [10]) to further improve the approximation factor.

Phase III: Greedy Improvement

- Apply Phase I and Phase II to an instance of the original problem where the facility cost is scaled up by a given factor $\Delta\geq 1$, and output a feasible solution.
- Assume the costs of the current solution are (F,C,P). Pick a facility i with cost f_i such that the ratio

$$(C+P-C_i-P_i-f_i)/f_i$$

is maximized, where C_i and P_i are the corresponding transportation cost and penalty cost if facility i was added to the current solution. If the ratio is positive, open facility i and repeat this step; otherwise, stop.

The greedy improvement procedure can improve the worst case bound of Algorithm TP, as shown next. We omit the proof here as the analysis is very similar to those in [7] and [10].

Lemma 5. *For any given (R_f, R_c, R_p)-approximation algorithm for (14), there is a $(R_f + \ln(\Delta), 1 + \frac{R_c - 1}{\Delta}, 1 + \frac{R_p - 1}{\Delta})$-approximation algorithm.*

By choosing $\alpha = 0.42539606, \delta = 0.17430753$, and $\Delta = 2.82899675$, we obtain the following theorem.

Theorem 3. *The URFLP with uniform failure probabilities admits a 2.674-approximation algorithm.*

5 Heuristics and Computational Tests

In this section, we propose several heuristics to solve the general URFLP: Sample Average Approximation Heuristic (SAA-H), Greedy Adding Heuristic (GAD-H), and Greedy Adding and Substitution Heuristic (GADS-H). In order to evaluate the performance of these three heuristics, we apply them to the URFLP with uniform failure probabilities. The reason is that the latter admits an integer programming formulation that can be solved to optimality by using commercial solvers such as CPLEX, so that we can compare the heuristic results with the exact solutions.

The test dataset is generated as follows. The coordinates of the sites are drawn from $U[0, 1] \times U[0, 1]$, demand of each site is drawn from $U[0, 1000]$ and rounded to the nearest integer, fixed facility costs are drawn from $U[500, 1500]$ and rounded to the nearest integer, and penalty costs are drawn from $U[0, 15]$. Further, the transportation cost c_{ij} is set to be the Euclidean distance between points i and j. The number of sites varies from 10 to 100.

All the algorithms were coded in C++ and tested on a Dell Optiplex GX620 computer running the Windows XP operating system with a Pentium IV 3.6 GHz processor and 1.0 GB RAM.

5.1 Sample Average Approximation-Heuristic

The sample average approximation method is widely used for solving complicated stochastic discrete optimization problems, e.g., [9], [13], and [22]. The basic idea of this method is to use a sample average function to estimate the expected value function. We apply the following procedures to solve the (URFLP-SP).

The SAA Heuristic

Step 1: Randomly generate a sample of N scenarios $\{A_1, \cdots, A_N\}$ and solve the following SAA problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in F} f_i y_i + \sum_{s=1}^{N} \frac{1}{N} \left(\sum_{j \in D} \sum_{i \in F} d_j c_{ij} x_{ij}^s + \sum_{j \in D} d_j r_j z_j^s \right) \\ \text{s.t.} \quad & \sum_{i \in F} x_{ij}^s + z_j^s = 1, \quad \forall j \in D, s = 1, \ldots N \\ & x_{ij}^s \leq y_i I_{A_s,i} \quad \forall i \in F, j \in D, s = 1, \ldots N \\ & y_i, x_{ij}^s, z_j^s \in [0,1] \end{aligned}$$

Repeat this step M times. For each $m = 1, 2, \cdots, M$, let y^m and v^m be the corresponding optimal solution and its optimal objective value respectively. In view of Proposition 1, the level assignment decision (the second stage decision) can be solely determined by y^m. Now, compute the true objective value $\hat{v}^m$ using the formulation of URFLP-NIP for each y^m, $m = 1, 2, \cdots, M$.

Step 2: Among the M solutions obtained in the first step, output the one with the minimum objective value, i.e., $\hat{v}^{min} = \min\limits_{m=1,\ldots,M} \hat{v}^m$.

Two remarks are in order. First, in a standard SAA approach (e.g. [13] and [22]), one more independent sample is needed to estimate the true expected value $\hat{v}^m$. But in our case, an analytical formula is used to estimate the true expected value. Second, the average of the v^m values, i.e. $\bar{v}^M = \sum_{m=1}^{M} v^m$, does not provide a lower bound for the optimal value. The reason is that different samples may lead to different solution spaces of the problem, which is due to the provider-side uncertainty. Nonetheless, $\bar{v}^M$ may still serve as a good indication of the quality of the solution from the SAA approach, as we will illustrate in the following computational tests.

We first test how the sample size (N) affects (1) the quality of the solution; and (2) the efficiency of the program, for a 50-node dataset with the uniform failure probabilities varying from 0 to 1. Table 1 lists the objective values obtained from SAA-H when $M = 1$ and the sample size varies from 10 to 50.

Failure Probability	N10	N20	N30	N40	N50	Exact
0	7197.27	7197.27	7197.27	7197.27	7197.27	7197.27
0.1	7956.03	7956.03	7763.8	7763.8	7763.8	7763.8
0.2	9429.49	8770.31	8425.99	8425.99	8425.99	8425.99
0.3	9908.52	10059.9	9669.49	10201.5	10095.2	9275.99
0.4	11546.9	11753.5	11984	13887.6	11937.6	10253.9
0.5	18727.5	15128.1	13120.6	15052.8	13546.4	11603
0.6	27429	18161.7	17946.4	17946.4	17946.4	13416.8
0.7	32965.3	33139.2	27374.8	23659.3	23690.4	16157.2
0.8	62876.9	46387.5	35743.6	35743.6	35743.6	21500.7
0.9	84406.1	69255.4	54722.3	54728.1	55647.4	35987.7
1	128009	128009	128009	128009	128009	128009

Table 1: Objective values from SAA-H for 50-node dataset

It is clear from Table 1 that the solution quality can be improved by increasing the sample size. The ratios of the objective value obtained from the SAA-H to the optimal value are plotted in Figure 1, and the computation times are plotted in Figure 2.

Figure 1 shows that the SAA-H with $M = 1$ obtains fairly good solutions when the failure probability is small ($p \leq 0.3$) but not very good solutions when the failure probability is big ($p > 0.3$). The following could be a possible explanation. When the failure probability is small, the majority of the facilities are candidate sites for opening in each sample, so the sets of candidate facilities are similar in different samples. Thus, any individual sample can capture the characteristics of the system pretty well, and the corresponding solution obtained from SAA-H is close to optimal. For the extreme case where $p = 0$, all facilities are available to open in each sample, so the sets of available facilities are the same in each sample and the SAA-H can produce the exact solution in

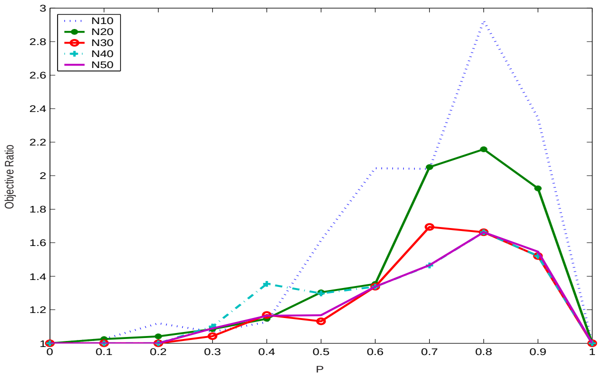

Figure 1: Objective ratio across different sample sizes for 50-node dataset

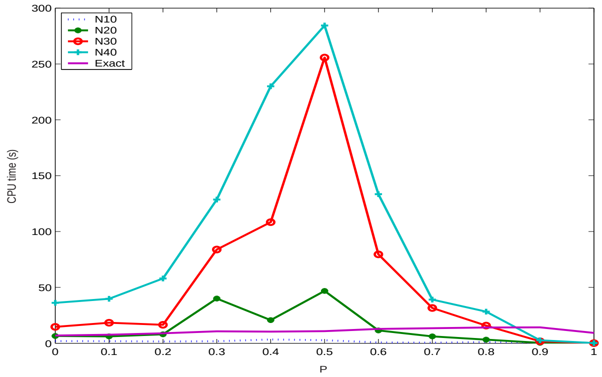

Figure 2: CPU time across different sample sizes for 50-node dataset

this case.

Figure 2 shows that the case with $N = 10$ is the only one that requires slightly less time than the exact algorithm, whereas others require more time. Another interesting pattern in Figure 2 is the tail effect of the CPU time in term of the failure probability. SAA-H spends more time to obtain a solution when the failure probability is around 0.5. One possible way to explain this phenomena is the following: when the failure probability is around 0.5, the constraints $x_{ij}^s \leq y_i I_{A_s,i}$ among different samples are quite different. As a result, the problem size increases, and so does the computation time.

Next, we examine the effect of the replication number (M) on the solution quality by fixing $N = 30$. Table 2 provides the objective values obtained when $M = 5, 10, 15, 20$. From the objective values obtained in different replication numbers, we can see that the increase of the replication number has not affected the solution quality too much. The gap in this table is defined as $\frac{\hat{v}^{min} - \bar{v}^M}{\bar{v}^M} \times 100\%$. The negative numbers in the "gap" column tell that $\bar{v}^M$ is not always a lower bound for $\hat{v}^{min}$. However, it is a good indication for the quality of the solution from SAA-H. In this particular case, if the gap is within $\pm 10\%$, the obtained objective value is close to the optimal value.

Failure	$M = 5$			$M = 10$			$M = 15$			$M = 20$			
Prob.	$\hat{v}^{min}$	$\bar{v}^{5}$	gap(%)	$\hat{v}^{min}$	$\bar{v}^{10}$	gap(%)	$\hat{v}^{min}$	$\bar{v}^{15}$	gap(%)	$\hat{v}^{min}$	$\bar{v}^{20}$	gap(%)	Exact
0	7197.27	7197.27	0.00	7197.27	7197.27	0.00	7197.27	7197.27	0.00	7197.27	7197.27	0.00	7197.27
0.1	7763.8	7687.03	1.00	7763.8	7760.14	0.05	7763.8	7784.17	-0.26	7763.8	7768.5	-0.06	7763.8
0.2	8425.99	8315.65	1.33	8425.99	8436.5	-0.12	8425.99	8484.24	-0.69	8425.99	8453.66	-0.33	8425.99
0.3	9414.4	9054.28	6.79	9378.06	9112.31	3.32	9378.06	9138.8	2.62	9275.99	9152.36	2.47	9275.99
0.4	10872.6	9740.45	23.03	10479.8	9814.09	10.79	10479.8	9826.75	6.65	10259.9	9842.41	6.48	10253.9
0.5	11932	10457	25.47	11932	10497.6	13.66	11932	10507.4	13.56	11932	10530.8	13.31	11603
0.6	17825.9	11377.6	57.73	17335.5	11475.9	55.33	17335.5	11523.5	50.44	17291.5	11563	49.92	13416.8
0.7	27157.4	12758.8	114.56	23227.4	12816.2	111.90	22894.2	12766.1	81.95	22894.2	12747.7	79.59	16157.2
0.8	34912.5	14758.3	142.19	31284.2	14761.9	136.50	31284.2	14736.8	112.29	31284.2	14746	112.15	21500.7
0.9	54722.3	19703.5	177.73	54628.8	19428.6	181.66	54628.8	19752	176.57	53343.9	19811.8	175.74	35987.7
1	128009	128009	0.00	128009	128009	0.00	128009	128009	0.00	128009	128009	0.00	128009

Table 2: Runs from SAA-H for 50-node dataset

Overall, SAA-H can not obtain very good solutions within a reasonable time, especially when the failure probability is fairly large. We defer presenting the computational results for the general case to Section 5.3, where we also compare the performances of the SAA-H and the greedy method.

5.2 Greedy Methods: GAD-H and GADS-H

In this section, we develop two related heuristics based on greedy local search and iterative improvement algorithms. In formulation (URFLP-NIP), Proposition 1 ensures that the level assignments can be easily derived for a given set of open facilities. Therefore, one can concentrate on selecting a set of open facilities without worrying too much on the level assignment decisions. Let $v(T)$ denote the objective function value given by the set of open facilities T. Let T^t be the set of open facilities at step t and Φ be the empty set.

Greedy Adding Heuristic

Step 1: The initial set of open facilities is empty. That is, set $t = 0$ and $T^t_{open} = \Phi$.

Step 2: Choose a facility from the remaining candidate facilities to open such that it can reduce the total cost the most. Add such facility to the open facility set. That is,
$t = t + 1$,
$j_t = \arg_{j \in F \setminus T^{t-1}} \min(v(T^{t-1} \cup \{j\}))$,
$T^t = T^{t-1} \cup \{j_t\}$.

Step 3: Repeat Step 2 until the current solution cannot be improved further.

In general, as we can see from the computational tests, the greedy adding heuristic is able to find a high quality solution very efficiently. Table 3 lists the computational results of 50-node dataset when the failure probability varies from 0 to 1. The first column, P, is the failure probability at each facility. The "gap" column is defined as the percentage difference between the cost of the solution obtained by GAD-H and the optimal cost.

As we can see from Table 3, GAD-H finds optimal or near-optimal solutions in most cases in less than 0.05 seconds. Comparing to the exact method using CPLEX, it spends much less time. The greedy adding algorithm seems to perform better when the facility failure probability is high.

P	Greedy Adding Heuristic		Exact Algorithm		
	Objective	Time (s)	Objective	Time (s)	gap(%)
0	7551.02	0.00	7197.27	6.94	4.92
0.1	8053.11	0.00	7763.80	7.61	3.73
0.2	8637.46	0.00	8425.99	8.94	2.51
0.3	9309.50	0.00	9275.99	10.62	0.36
0.4	10253.90	0.02	10253.90	10.38	0.00
0.5	11622.80	0.02	11603.00	10.86	0.17
0.6	13416.80	0.02	13416.80	12.84	0.00
0.7	16157.20	0.02	16157.20	13.47	0.00
0.8	21500.70	0.02	21500.70	14.08	0.00
0.9	35987.70	0.05	35987.70	14.27	0.00
1	128009.00	0.00	128009.00	9.27	0.00

Table 3: 50-node Uniform Case: Greedy Adding and Exact Method

It actually finds optimal solutions when the failure probability exceeds 0.5. This is in contrast to the performance of the SAA-H, which works better when the failure probability is low.

After the greedy adding heuristic, we perform the following greedy substitution heuristic to further improve the solution: at each iteration, a substitute facility is chosen to replace the existing open facility if doing so reduces the total cost the most. This procedure is repeated until no substitute facility can be found to reduce the total cost. The substitution can be a null facility. Replacing an open facility with a null facility means that we close the facility. After the substitution process, another greedy adding procedure is performed to further improve the solution. The whole process (a greedy adding procedure followed by a greedy substitution procedure then followed by another greedy adding procedure) is called the greedy adding and substitution heuristic (GADS-H).

Our computational experiments show that GADS-H is much better algorithm compared with GAD-H. It actually finds the optimal solutions for all instances in Table 3 and the CPU times are comparable with those reported by GAD-H. The results are summarized in Table 4.

It is interesting to compare the sets of open facilities in Table 4. One might conclude that more

P	GADS-H		Exact Algorithm			Gap
	Objective	Time (s)	Objective	Open Facilities	Time (s)	(%)
0	7197.27	0.02	7197.27	15 31 40 41 48	6.94	0.00
0.1	7763.80	0.02	7763.80	15 22 31 40 41 48	7.61	0.00
0.2	8425.99	0.03	8425.99	15 22 31 40 41 48	8.94	0.00
0.3	9275.99	0.02	9275.99	15 22 31 40 41 48	10.62	0.00
0.4	10253.90	0.02	10253.90	15 22 31 40 41 42 48	10.38	0.00
0.5	11603.00	0.03	11603.00	15 22 31 35 40 41 42 48	10.86	0.00
0.6	13416.80	0.06	13416.80	2 12 15 22 31 35 40 41 43 48	12.84	0.00
0.7	16157.20	0.09	16157.20	2 12 14 15 19 22 31 35 40 41 43 45 48	13.47	0.00
0.8	21500.70	0.20	21500.70	2 12 14 15 19 20 21 22 26 31 35 36 40 41 43 45 48	14.08	0.00
0.9	35987.70	0.48	35987.70	1 2 10 11 12 14 15 17 19 20 21 22 23 24 26 27 31 35 36 40 41 43 45 48 49	14.27	0.00
1	128009.00	0.00	128009.00	No open facility	9.27	0.00

Table 4: 50-node Uniform Case: GADS-H and Exact Solution

facilities should be open as the facilities get more vulnerable, that is, when the failure probability increases. Although this claim is usually valid, it is not always true. An extreme case is when failure probability is one so that no facility should open. One can also consider the following counter example where there is only one single facility to open. If $f_1 + d_{11}r_1p_1 < d_{11}r_1$, then this facility should be open. If p_1 increases to p_1' such that $f_1 + d_{11}r_1p_1' > d_{11}r_1$, then this facility should not be open. In this example, When the failure probability increases, fewer facilities are open in this example.

5.3 General Case: SAA-H, GAD-H and GADS-H

Dropping the uniformity assumption on the facility failure probabilities introduces more challenges to solve the URFLP exactly, due to the nonlinearity in formulation (URFLP-NIP). In this section, we apply SAA-H, GAD-H and GADS-H to the general problem and compare their performance. Several datasets are derived from the 100-node data set. Table 5 lists the objective values obtained by these three heuristics and their computational times. The table only contains results from a one-run SAA-H with a sample size of $N = 100$. Multiple runs of SAA-H shows little improvement on the solution.

Dataset	GAD-H		GADS-H		SAA-H ($N = 100$, $M = 1$)	
#	Objective	Time (s)	Objective	Time (s)	Objective	Time (s)
node-20	5761.79	0.00	5761.79	0.00	5761.79	3.09
node-30	7420.86	0.00	7382.04	0.00	7622.22	50.67
node-40	7474.92	0.00	7474.92	0.02	7474.92	54.20
node-50	8763.75	0.00	8641.28	0.00	8641.28	185.56
node-60	9357.37	0.00	9357.37	0.02	9394.87	159.48
node-70	10337.60	0.00	10337.60	0.03	10391.80	250.85
node-80	11054.30	0.00	11054.30	0.05	11054.30	320.82
node-90	13030.90	0.00	12405.50	0.06	12405.50	659.57
node-100	14463.40	0.02	13820.87	0.09	14028.10	2164.03

Table 5: General Case: SAA-H vs. Greedy Methods

¿From Table 5 we can see that GADS-H finds the best solutions among all three heuristics, and its CPU times are comparable with those using GAD-H. For some instances, SAA-H also find the best solutions, but it spends much more time compared with GADS-H. Based on these observations, we conclude that among the three heuristics we proposed, GADS-H is the most suitable heuristic for the URFLP.

6 Conclusions

In this paper, we propose and analyze two formulations for the reliable facility location problem, where every open facility can fail with a certain probability. If a facility fails, customers originally assigned to it have to be reassigned to other facilities that are operational. This problem is NP-hard and we propose a 2.674-approximation algorithm for the uniform case where all facilities have the same failing probability. We also propose several heuristics that can produce near-optimal solutions for the general problem.

There are several interesting future research directions. We note that the main limitation of the current models is the assumption that the facilities are uncapacitated. Although the assumption itself is very common in the facility location models, it may be unrealistic in practice. In the capacitated case, customer of failed facilities can be assigned to the next level backup facilities only if they have sufficient capacity to satisfy the additional demand. This makes the capacitated model very complex. We plan to investigate this direction in the near future. In addition, some new measurements of the reliability concept in the facility location problem settings are worth pursuing. Finally, we think it is important to design efficient and powerful meta heuristics tailored for the complex reliable facility location problems.

References

[1] O. Berman, D. Krass, and M.B.C. Menezes. Facility reliability isuues in network p-median problems: strategic centralization and co-location effects. *Operations Research*, 2006. To appear.

[2] A. F. Bumb and J.C.W. van Ommeren. An approximation algorithm for a facility location problem with stochastic demands and inventory. *Operations Research Letters*, 34(3):257–263, 2006.

[3] J. Byrka. *An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem*, 2007. Working paper.

[4] M. S. Daskin. *Network and Discrete Location: Models, Algorithms, and Applications*. Wiley, New York, 1995.

[5] M.S. Daskin, K. Hogfan, and C. ReVelle. α-reliable p-minimax regres: a new model for strategic facility locaton modeling. *Location Science*, 5(4):227–246, 1997.

[6] P.M. França and H.P.L. Luna. Soving stochastic transportation-location problems by generalized benders decomposition. *Transportatoin Science*, 16(2):113–126, 1982.

[7] S. Guha, A. Meyerson, and K. Munagala. A constant factor approximation algorithm for the fault-tolerant facility location problem. *Journal of Algorithms*, 48:429–440, 2003.

[8] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani. An improved approximation algorithm for uncapacitated facility location problem with penalties. *Journal of the ACM*, 50:795–824, 2003.

[9] A. J. Kleywegt, A. Shapriro, and T. Homem-de mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal of Optimization*, 12(2):479–502, 2001.

[10] M. Mahdian, Y. Ye, and J. Zhang. Approximation algorithms for metric facility location problems. *SIAM Journal on Computing*, 36:411–432, 2006.

[11] R. Mirchandani and R. Francis, editors. *Discrete Location Theory*. Wiley, New York, 1990.

[12] S. H. Owen and M. S. Daskin. Strategic facility location: A review. *European Journal of Operational Research*, 111:423–447, 1998.

[13] T. Santoso, S. Ahmed, M. Goetshalckx, and A. Shapriro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167:96–115, 2005.

[14] P. M. Scaparra and R. L. Church. A bilevel mixed integer program for critical infrastructure protection planning. *Computers and Operations Research*.

[15] P. M. Scaparra and R. L. Church. *An optimal approach for the interdiction median problem with fortification*, 2005. Working paper, Kent business School.

[16] Z. J. Shen. A profit maximizing supply chain network design model. *Operations Research Letters*, 34:673–682, 2005.

[17] D. Shmoys, É. Tardos, and K. Aasdal. Approximation algortihms for facility locatoin problems. In *Proceedings of 29th ACM STOC*, pages 265–274, 1997.

[18] L. V. Snyder. Facility location under uncertainty: a review. *IIE Transactions*, 38:537–554, 2006.

[19] L. V. Snyder and M.S. Daskin. Reliability models for facility location: The expected failure cost case. *Transportation Science*, 39:400–416, 2005.

[20] L.V. Snyder, P. M. Scaparra, M. S. Daskin, and R. L. Church. Planning for disruptions in supply chain networks. In *TutORials in Operations Research*. INFORMS, 2006.

[21] C. Swamy and D. Shmoys. Approximation algorithms for 2-stage stochastic optimization problems. *ACM SIGACT News*, 37(1):33–46, 2006.

[22] B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, and A. Shapriro. The sample average approximation method applied to stochastic routing problems: a computational study. *Computational Optimization and Applications*, 24:289–333, 2003.

[23] T. Zeng and J.E. Ward. The stochastic location-assignment problem on a tree. *Annals Operations Reseaerch*, 136:81–97, 2005.

A Appendix: Proofs of the Propositions and Theorems

A.1 Formulation

Proof. Proof of Theorem 1.

(1). Assume we have an optimal solution (x, y, z) for problem (13), then we can construct a feasible solution $(\bar{x}, \bar{y}, \theta)$ for problem (14) with the same objective value.

Let $\bar{x} = x$ and $\bar{y} = y$. For each $j \in D$ and $k \geq 1$, define $\theta_j^k = \sum_{t=1}^k z_j^t$. Because for each j, there is exactly one value of z_j^t would be equal to 1, we can conclude $\theta_j^k \in \{0, 1\}$. And it is straightforward that $(\bar{x}, \bar{y}, \theta)$ is a feasible solution to problem (14). Now we check the objective value corresponding to $(\bar{x}, \bar{y}, \theta)$. In fact, we only need to consider

$$\begin{aligned}
\sum_{k=1}^{\infty} p^{k-1}(1-p)\theta_j^k &= \sum_{k=1}^{\infty} p^{k-1}(1-p) \sum_{t=1}^{\min\{k,|F|+1|\}} z_j^t \\
&= \sum_{t=1}^{|F|+1} \sum_{k=t}^{\infty} p^{k-1}(1-p) z_j^t \\
&= \sum_{t=1}^{|F|+1} p^{t-1} z_j^t
\end{aligned}$$

The first equality is due to the fact that $\forall t > |F| + 1, j \in D, z_j^t = 0$.

(2). Now we prove the other direction; i.e., assume we have an optimal solution (x, y, θ) for problem (14), then we can construct a feasible solution $(\bar{x}, \bar{y}, z)$ for problem (13) with the same objective value.

To that end, we first show it is without losing of generality to assume that for each $j \in D$, if $\theta_j^k = 1$, then $\theta_j^{k+1} = 1$. Otherwise, assume there exist j_0 and k_0 such that $\theta_{j_0}^{k_0} = 1$ but $\theta_{j_0}^{k_0+1} = 0$. Then from the constraints, we know that $x_{ij_0}^{k_0} = 0$ for all i and there exists an i_0 such that $x_{i_0 j_0}^{k_0+1} = 1$.

Now we define a new solution $(\tilde{x}, \tilde{y}, \tilde{\theta})$ for problem (14) as follows: $\tilde{y} = y$; for $j \neq j_0$, $\tilde{x}_{ij}^k = x_{ij}^k$, $\tilde{\theta}_j^k = \theta_j$ for all $i \neq i_0$ and k; for $k \neq k_0, k_0+1$, $\tilde{x}_{ij_0}^k = x_{ij_0}^k$, $\tilde{\theta}_{j_0}^k = \theta_{j_0}$ for all $i \neq i_0$; $x_{i_0 j_0}^{k_0} = 1$, $\theta_{j_0}^{k_0} = 0$, $x_{i_0 j_0}^{k_0+1} = 0$ and $\theta_{j_0}^{k_0+1} = 1$. It is easy to verify that $(\tilde{x}, \tilde{y}, \tilde{\theta})$ is a feasible solution for problem (14). The difference between the objective value corresponding to (x, y, θ) and the objective value corresponding to $(\tilde{x}, \tilde{y}, \tilde{\theta})$ is

$$\Big(d_{j_0} c_{i_0 j_0} p^{k_0}(1-p) + d_{j_0} r_{j_0} p^{k_0-1}(1-p)\Big) - \Big(d_{j_0} c_{i_0 j_0} p^{k_0-1}(1-p) + d_{j_0} r_{j_0} p^{k_0}(1-p)\Big)$$

which should be less than or equal to zero given that (x, y, θ) is an optimal solution to problem (14). It follows that

$$r_{j_0} \leq c_{i_0 j_0}.$$

If we define yet another solution $(\hat{x}, \hat{y}, \hat{\theta})$ such that it is the same as (x, y, θ) with the following exceptions:

$$\hat{x}_{i_0 j_0}^{k_0} = 0, \hat{\theta}_{j_0}^{k_0} = 1, \hat{x}_{i_0 j_0}^{k_0+1} = 0, \hat{\theta}_{j_0}^{k_0+1} = 1.$$

Again, this solution it feasible. Also, the difference between the objective value corresponding to (x, y, θ) and the objective value corresponding to $(\hat{x}, \hat{y}, \hat{\theta})$ is

$$d_{j_0} c_{i_0 j_0} p^{k_0}(1-p) - d_{j_0} r_{j_0} p^{k_0}(1-p) \geq 0,$$

which implies that $(\hat{x}, \hat{y}, \hat{\theta})$ is an optimal solution for problem (14) and it satisfies the condition that if $\hat{\theta}_{j_0}^{k_0} = 1$ then $\hat{\theta}_{j_0}^{k_0+1} = 1$.

Therefore, we can assume that for each $j \in D$, if $\theta_j^k = 1$, then $\theta_j^{k+1} = 1$. Now we are ready to find a feasible solution for problem (13). Let $\bar{x} = x, \bar{y} = y$. And for each j, if $\theta_j^k = 1$ and $\theta_j^{k-1} = 0$, then let $z_j^k = 1$ and let $z_j^{k'} = 0$ for $k' \neq k$; if $\theta_j^1 = 1$, then let $z_j^1 = 1$ and $z_j^{k'} = 0$ for $k' \geq 2$. Then it is clear from the definition that $\theta_j^k = \sum_{t=1}^k z_j^t$. Then we know that $(\bar{x}, \bar{y}, z)$ is feasible for problem

(13). And from the first part of the proof, we know the objective value corresponding to $(\bar{x}, \bar{y}, z)$ is the same as the one corresponding to (x, y, θ). This completes the proof. $\square$

A.2 Approximation Algorithms of the Uniform Case

Proof. Lemma 3. We bound the expected penalty cost for each client $j \in D$, which depends on the value of β that is randomly generated. We consider two cases.

Case 1. $\theta_j^{k_j} \geq \beta$. Then client j is assigned to exactly $m_j = k_j - 1$ facilities. Therefore, a penalty cost will incur if all of these $(k_j - 1)$ facilities failed, which happens with probability p^{k_j-1}. Therefore, the expected total penalty cost for client j is $d_j r_j p^{k_j-1}$. Notice that $\theta_j^k = 1$ for all $k \geq k_j + 1$. Then

$$\begin{aligned} d_j r_j p^{k_j-1} &= \sum_{k=k_j}^{|F|} d_j r_j p^{k-1}(1-p) + \sum_{k=|F|+1}^{\infty} d_j r_j p^{k-1}(1-p) \\ &= d_j r_j p^{k_j-1}(1-p) + \sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k \end{aligned}$$

Case 2. $\theta_j^{k_j} < \beta$. In this case, client j is assigned to exactly $m_j = k_j$ facilities. A penalty cost will incur if all of these k_j facilities failed, which happens with probability p^{k_j}. Therefore, the expected total penalty cost for client j is $d_j r_j p^{k_j}$, which is equal to

$$\sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p) = \sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k.$$

If $\theta_j^{k_j} \geq \delta$, then Case 1 happens with probability 1. Thus the expected penalty cost of client j is

$$d_j r_j p^{k_j-1}(1-p) + \sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k \leq \frac{1}{\delta} d_j r_j p^{k_j-1}(1-p)\theta_j^{k_j} + \sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k.$$

On the other hand, if $\theta_j^{k_j} < \delta$, then Case 1 happens with probability $\frac{\theta_j^{k_j}}{\delta}$, and thus Case 2 happens with probability $1-\frac{\theta_j^{k_j}}{\delta}$. Therefore, the expected penalty cost of client j is bounded by

$$\begin{aligned}
& \frac{\theta_j^{k_j}}{\delta}\left(d_j r_j p^{k_j-1}(1-p)+\sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k\right)+(1-\frac{\theta_j^{k_j}}{\delta})\sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k \\
= \; & \frac{1}{\delta} d_j r_j p^{k_j-1}(1-p)\theta_j^{k_j}+\sum_{k=k_j+1}^{\infty} d_j r_j p^{k-1}(1-p)\theta_j^k.
\end{aligned}$$

This completes the proof. □

Proof. Lemma 4. The costs of interest depend on the value of the randomly generated β. Recall the definition of $(\hat{x},\hat{y})$. For each $j\in D$ and $i\in F$, $\hat{x}_{ij}^k=x_{ij}^k$, if $k\neq m_j$; If $m_j=k_j-1$, then $\hat{x}_{ij}^{k_j}=0\leq x_{ij}^{k_j}$; if $m_j=k_j$, then $\hat{x}_{ij}^{k_j}=\frac{x_{ij}^{k_j}}{1-\theta_j^{k_j}}$. The latter is true only if $\theta_j^{k_j}<\beta$. Therefore, in both cases,

$$\hat{x}_{ij}^k\leq\frac{1}{1-\beta}x_{ij}^k. \tag{19}$$

By the constraint of the linear programming relaxation of (14), we know that, for each $i\in F$ and $j\in D$,

$$\sum_{k=1}^{|F|} x_{ij}^k\leq y_i.$$

Therefore,

$$\hat{y}_i=\max_{j\in D}\sum_{k=1}^{k_j}\hat{x}_{ij}^k\leq\max_{j\in D}\frac{1}{1-\beta}\sum_{k=1}^{k_j}x_{ij}^k\leq\frac{1}{1-\beta}y_i. \tag{20}$$

To show that

$$\sum_{i\in F}\hat{x}_{ij}^k=1 \quad \forall j\in D, k\leq m_j,$$

we consider the following two cases.

- $m_j=k_j-1$: In this case, $\forall j\in D, k\leq m_j$, $\theta_j^k=0$ and $\hat{x}_{ij}^k=x_{ij}^k$. From the constraint 16, $\displaystyle\sum_{i\in F}\hat{x}_{ij}^k=\sum_{i\in F}x_{ij}^k=1$.
- $m_j=k_j$: when $k<k_j$, from the proof of the first bullet, we know that $\sum_{i\in F}\hat{x}_{ij}^k=1$. when $k=m_j$, $\displaystyle\sum_{i\in F}\hat{x}_{ij}^{k_j}=\frac{\sum_{i\in F}x_{ij}^k}{1-\theta_j^{k_j}}=1$. The last equality holds because of the constraint 16, i.e. $\displaystyle\sum_{i\in F}x_{ij}^k+\theta_j^{k_j}=1$.

Thus $\hat{x}$ and $\hat{y}$ satisfy the following constraints:

$$\sum_{i\in F} \hat{x}_{ij}^k = 1 \quad \forall j \in D, k \leq m_j$$

$$\sum_{k=1}^{m_j} \hat{x}_{ij}^k \leq \hat{y}_i \quad \forall i \in F, j \in D$$

$$\hat{x}_{ij}^k \geq 0,$$

$$0 \leq \hat{y}_i \leq 1.$$

Therefore, if we apply the algorithm in [7] with a parameter α to round the solution $(\hat{x}, \hat{y})$, then by Lemma 1, we can construct a solution $(\bar{x}, \bar{y})$ such that the expected total facility cost and total transportation cost is bounded above by

$$\begin{aligned} & \frac{\ln\frac{1}{\alpha}}{1-\alpha}\sum_{i\in F} f_i\hat{y}_i + \frac{3}{1-\alpha}\sum_{j\in D}\sum_{k=1}^{m_j}\sum_{i\in F} d_j c_{ij}\hat{x}_{ij}^k p^{k-1}(1-p) \\ \leq \quad & \frac{1}{1-\beta}\left(\frac{\ln\frac{1}{\alpha}}{1-\alpha}\sum_{i\in F} f_i y_i + \frac{3}{1-\alpha}\sum_{j\in D}\sum_{k=1}^{m_j}\sum_{i\in F} d_j c_{ij} x_{ij}^k p^{k-1}(1-p)\right) \\ \leq \quad & \frac{1}{1-\beta}\left(\frac{\ln\frac{1}{\alpha}}{1-\alpha}\sum_{i\in F} f_i y_i + \frac{3}{1-\alpha}\sum_{j\in D}\sum_{k=1}^{k_j}\sum_{i\in F} d_j c_{ij} x_{ij}^k p^{k-1}(1-p)\right). \end{aligned}$$

The first inequality holds because of inequalities 19 and 20; the second one holds because of $m_j \leq k_j$.

Finally notice that β was uniformly distributed in $(0, \delta)$, thus

$$\mathrm{E}[\frac{1}{1-\beta}] = \int_0^\delta \frac{1}{1-\beta}\frac{1}{\delta}d\beta = \frac{1}{\delta}\ln\frac{1}{1-\delta}.$$

This completes the proof of the Lemma. □