An Efficient Approach for Solving Reliable Facility Location Models

Robert Aboolian

College of Business Administration, California State University San Marcos, San Marcos, California 92096,
raboolia@csusm.edu

Tingting Cui

Department of Industrial Engineering and Operations Research, University of California Berkeley, Berkeley, CA 94720,
tingting@ieor.berkeley.edu

Zuo-Jun Max Shen

Department of Industrial Engineering and Operations Research, University of California Berkeley, Berkeley, CA 94720,
shen@ieor.berkeley.edu

We consider reliable facility location models in which facilities are subject to unexpected failures and customers may be re-assigned to facilities other than their regular facilities. The objective is to minimize the total expected costs in normal and failure scenarios. We allow facilities to have different failure rates and do not limit the number of facilities that might be assigned to a customer. Lower bounds for Reliable Uncapacitated Fixed-charge Location Problem (RUFLP) are derived and used to introduce a class of efficient algorithms for solving the RUFLP problem.

1. Introduction

Consider a supply network design problem in which the facilities, once opened, are subject to unexpected failures, and its customers may have to be served by facilities further than their preferred locations. Such situations are commonly seen in practice, where the facility failures may be originated from natural disasters, labor actions, or terrorist attacks. This brings up the need to design and operate reliable supply chains that are resilient to all sorts of disruptions.

Risk of failure at each facility is specified by an individual and independent failure probability inherent to that facility. To hedge against failures within a supply network, each customer is assigned to multiple facilities, ordered by levels. The lowest level facility assigned to a customer is the primary facility to serve that customer so long as it has not failed and remains operational. A higher level facility assigned to a customer works as a backup facility and serves that customer only if all facilities at lower levels have failed. At

1

the rare occasion in which all facilities assigned to a customer have failed, a penalty cost is incurred. The penalty cost can be taken as lost sales, loss of goodwill, or the cost to serve the customer at a competitor's facility.

In this paper, we study the reliable version of the *uncapacitated fixed-charge location problem* ($UFLP$), but our model can be easily extended to address other facility location problems. Here, the objective is to minimize the sum of fixed location costs, the expected transportation costs (at all levels), and the expected penalty costs. This problem will be referred to as the *reliable uncapacitated fixed-charge location problem* ($RUFLP$).

When compared with cost minimizing supply network design, the reliable facility networks, in general, tend to require more facilities for backup solutions. While adding facilities increases the system's overall fixed cost, it helps to hedge against the excessive transportation and penalty costs in case of failures within the network.

Note that our problem should not be confused with the "k-level facility location problem", where each customer must be served by a sequence of k different kinds of facilities located in k levels of hierarchy (see Sahin and Sural (2007)).

Our work is related to the emerging literature of facility location under uncertainty. Some of the earlier literature focuses on demand uncertainty, motivated by facility congestion in emergency service systems. This includes Daskin (1982, 1983), Batta et al. (1989), Ball and Lin (1993) among others. Also see Daskin et al. (1988) for a survey of covering models under demand uncertainty.

Snyder and Daskin (2005) studied the facility reliability issues with a different motivation, where uncertainty comes from the supply side, or more specifically, the disastrous facility disruptions. In Snyder and Daskin (2005), the reliability models of $UFLP$ and the P-median problem were introduced and a multi-objective formulation was used to demonstrate trade off between the cost and reliability. Assuming that all facilities have identical failure probability, the authors formulated the problem as a linear mixed integer program, and employed Lagrangian relaxation for efficient solutions. The uniform failure probability assumption is also taken in other related papers on disruption management, including Snyder and Daskin (2006) and Lim et al. (2009).

Facility location in context of facility failure has also been studied in the "fault-tolerant facility location problem" literature. In this problem, to hedge against facility failures, each demand point is supposed to be assigned to a certain number of distinct facilities, but unlike

$RUFLP$, the expected transportation cost for the network is independent from the risk of failure in each facility. There are many papers in "fault-tolerant facility location problem" literature, which offer approximate solution approaches. These include Guha et al. (2003), Byrka et al. (2010), Swamy and Shmoys (2003), Yan and Chrobak (2011), and Chechik and Peleg (2010) among others. Facility location in the context of incurring a penalty cost when a customer is not served has also been studied in the "facility location problem with outliers". See Charikar et al. (2001) for approximate approaches for this problem.

Most recently, Z.-J.M.Shen et al. (2011), Berman et al. (2007), Cui et al. (2010) relax the assumption of identical failure probabilities. Z.-J.M.Shen et al. (2011) formulated the reliable $UFLP$ problem as a nonlinear mixed integer program and provided several heuristic solution algorithms. Berman et al. (2007) focused on asymptotic properties of the optimal solutions, along with heuristic algorithms with bounds on the worst-case performance. Cui et al. (2010) is the first to provide exact solutions for the $RUFLP$ problem with site-dependent failure probabilities, in which the problem is formulated as a linear mixed integer program and solved using Lagrangian relaxation. Lagrangian relaxation was originally proposed for the $UFLP$ by Geoffrion (1974) and for the P-median problem by Cornuejols et al. (1977). The Lagrangian relaxation algorithm in Cui et al. (2010) is efficient when the maximum number of facilities assigned to a customer is relatively small. However, for a large number of facilities assigned to a customer, the Lagrangian process fails to solve even moderate sized problem instances within a reasonable time frame.

Our paper can be taken as an extension of Cui et al. (2010), where we do not limit the number of facilities assigned to a customer. Using a novel approach, which is combination of neighborhood search and cutting plane process, our algorithm outperforms Cui et al. (2010) in both execution time and solution quality, especially when large number of facilities are allowed to be assigned to a customer. Our algorithm can also work as a heuristic, which can solve extremely large problem instances.

Local search heuristics have been used to solve facility location problems (e.g. see Ghosh (2003) and Aboolian et al. (2009a)), but even though they are proven to be very efficient in terms of computation time and solution quality, they fail to guarantee the exactness of the solution. In this paper, we integrate a local search method with a cutting plane process to find the optimal solution for $RUFLP$ in a very efficient manner. A similar cutting plane process were previously used in Aboolian et al. (2008), Aboolian et al. (2009b) and

Aboolian et al. (2010), but to the best of our knowledge, this is the first paper to use a combination of a neighborhood search and a cutting plane process to solve a non-linear integer program, and we believe that this exact solution approach is quite novel. We also believe that this approach can be applied to many other facility location problems which can be modeled as non-linear integer programs.

The remainder of the paper is organized as follows. We introduce the formulation of the $RUFLP$ in Section 2, and describe solution algorithms for $RUFLP$ in Section 3. Section 3.1 introduces a linear reformulation that provides a lower bound, Section 3.2 proposes an approximate solution, and Section 3.3 discusses an exact algorithm. We then discuss our numerical experiment design and computational results in Section 4. Finally we provide concluding remarks in Section 5.

2. Formulation

Let $N(|N| = n)$ be the set of customer demand aggregation points and $M(|M| = m)$ be the set of candidate locations for the facilities. We denote demand rate at node i as λ_i for each $i \in N$, the fixed cost to locate a facility at node j as f_j for each $j \in M$. Let c_{ij} be the shipping cost of one unit of demand from a facility located at node j to the customers at node i. We model facility disruptions as independent events, happening at location $j \in M$ with probability $0 \le q_j < 1$. We will use $S \subset M$ to denote the set of facilities selected. If a facility is located at site j, we call it facility $j \in S$.

To hedge against disruption risk, each customer can be assigned and served by any open facility. A penalty cost ϕ_i is incurred for each unit of unmet demand due to facility failures, which can be taken as the lost of good will, or the cost to serve the customers at a competitor's facility. Although each customer can be assigned and served at any open facility, it makes no sense to serve a customer at a facility where its unit shipping cost is higher than the penalty cost. Let $S[i] = \{k \mid \phi_i \ge c_{ik}, k \in S\}$ and $s_i = |S[i]|$. Define $i[r, S]$ $\in S$ $(r = 1, 2, .., s_i)$ to be the facility in S that serves customers at $i \in N$ at level r. Define $C_i(S)$ to be the expected sum of shipping cost and lost sales cost of one unit of demand to customers at $i \in N$. Then, it can be verified that

$$C_i(S) = \left(1 - q_{i[1,S]}\right) c_{i,i[1,S]} + \sum_{r=2}^{s_i} \left(\prod_{t=1}^{r-1} q_{i[t,S]}\right) \left(1 - q_{i[r,S]}\right) c_{i,i[r,S]} + \prod_{t=1}^{s_i} q_{i[t,S]} \phi_i. \quad (1)$$

Define $F(S)$ to be the total fixed location and shipping cost given location set S, such that

$$F(S) = \sum_{j \in S} f_j + \sum_{i \in N} \lambda_i C_i(S). \tag{2}$$

The Reliable Uncapacitated Fixed-charge Location problem $(RUFLP)$ is formulated as:

$$\min_{S \subseteq M} \{F(S)\}.$$

Define $M[i] = \{j \mid \phi_i > c_{ij}, j \in M\}$ to be the set of facilities in M whose unit shipping cost to customers at $i \in N$ is lower than the unit lost sales cost ϕ_i. Let $m_i = |M[i]|$. To include the unit lost sales cost ϕ_i in the cost calculations, we use a "dummy" facility, denoted by J, that has a fixed cost $f_J = 0$, a failure probability $q_J = 0$ and a transportation cost $c_{iJ} = \phi_i$ for customer $i \in N$. A "level-r" assignment for a customer $i \in N$ will serve her if and only if all of her assigned facilities at levels $1, ..., r-1$ have failed. To capture the possibility of failure for all regular facilities, at the optimal solution, customer $i \in N$ at the last level must be assigned to the dummy facility J. Denote x_j to be a binary variable which is one if we open a facility at j and zero otherwise. Denote y_{ijr} to be a binary variable which is one if facility j is assigned to customers at i at level r and zero otherwise. Finally, let $P_{ijr} y_{ijr}$ be the probability that facility j serves customers i at level r given her other assigned facilities at levels 0 to $r-1$. Note that $P_{ij1} y_{ij1}$, the probability that facility j serves customer i at level 1, is just the probability that j remains open, so $P_{ij1} = 1 - q_j$. For $2 \le r \le m_i + 1$, we have $P_{ij2} = (1-q_j) \sum_{k \in M[i]} q_k y_{i,k,1} = (1-q_j) \sum_{k \in M[i]} \frac{q_k}{1-q_k} P_{i,k,1} y_{i,k,1}$, $P_{ij3} = (1-q_j) \sum_{k \in M[i]} q_k y_{i,k,1} \sum_{l \in M[i]} q_k y_{i,l,2} = (1-q_j) \sum_{k \in M[i]} \frac{q_k}{1-q_k} P_{i,k,2} y_{i,k,2}$, and continuing the same pattern, we obtain $P_{ijr} = (1-q_j) \sum_{k \in M[i]} \frac{q_k}{1-q_k} P_{i,k,r-1} y_{i,k,r-1}$. Given the above definitions the reliable uncapacitated facility location problem $RUFLP$ can be rewritten as:

$$(RUFLP) \quad \text{Minimize } Z = \sum_{j \in M} f_j x_j + \sum_{i \in N} \sum_{j \in M[i] \cup \{J\}} \sum_{r=1}^{m_i+1} \lambda_i c_{ij} P_{ijr} y_{ijr} \tag{3}$$

$$\text{s.t.} \sum_{r=1}^{m_i} y_{ijr} \leq x_j \ \forall i \in N, \ j \in M[i], \tag{4}$$

$$\sum_{r=1}^{m_i+1} y_{iJr} = 1 \ \forall i \in N, \tag{5}$$

$$\sum_{j \in M[i] \cup \{J\}} y_{ijr} + \sum_{g=1}^{r-1} y_{iJg} = 1 \ \forall i \in N, \ 1 \leq r \leq m_i+1, \tag{6}$$

$$P_{ij1} = 1 - q_j \ \forall i \in N, \ j \in M[i] \cup \{J\}, \tag{7}$$

$$P_{ijr} = (1 - q_j) \sum_{k \in M[i]} \left(\frac{q_k}{1 - q_k} P_{i,k,r-1} y_{i,k,r-1} \right) \tag{8}$$

$$\forall i \in N, \ j \in M[i] \cup \{J\}, \ 2 \leq r \leq m_i+1,$$

$$x_j, y_{ijr} \in \{0, \ 1\} \ \forall i \in N, \ j \in M[i] \cup \{J\}, \ 1 \leq r \leq m_i+1. \tag{9}$$

The objective function (3) is the sum of the expected transportation costs and the fixed costs. Constraints (4) ensure that customers are only assigned to the open facilities, while constraints (5) enforce each customer has to be assigned to the dummy facility at a certain level. Constraints (6) require that for each customer i and each level r, either i is assigned to a facility at level r or it is assigned to the dummy facility J at certain level $g < r$ (taking $\sum_{g=1}^{r-1} y_{iJg} = 0$ if $r = 1$). Constraints (7) and (8) are the "transitional probability" equations. Note that constraints (6) imply that $y_{i,k,r-1}$ can equal to 1 for at most one $k = J$, which guarantees the correctness of the transitional probabilities.

Note that $RUFLP$ is a nonlinear mixed integer program which is large-scale in nature. To efficiently solve this problem Cui et al. (2010) proposed a linearized formulation and solved it using Lagrangian relaxation. The difference between our model and the model developed in Cui et al. (2010) is that in Cui et al. (2010) it is assumed that each customer can be assigned and served by up to $R \geq 1$ facilities, while in our model R, the maximum number of facilities assigned to a customer, is not limited to a certain fixed value and customers are able to be assigned at all open facilities with a shipping cost less than the penalty cost, which in fact makes the model more realistic. When R is fixed the above model can be rewritten by replacing $\min\{R, m_i\}$ in place of m_i for $i \in N$.

Here we develop a solution approach which finds an optimal solution more efficiently regardless of whether R is fixed or not. To do this we first simplify the customer assignment assumption such that customers are assigned to open facilities level by level in an increasing

order of shipping cost. This might not be the optimal customer assignment, but later we show that even with this assumption our solution approach is frequently able to come up with better solutions and with less amount of time compared to the Lagrangian relaxation algorithm developed in Cui et al. (2010). We also develop an efficient approximate approach which is capable of solving large problem instances.

3. Algorithms for $RUFLP$

Theexact and the approximate solution approaches developed in this Section are based on obtaining efficient lower and upper bounds for $RUFLP$. In Section 3.1, we develop a mixed integer program called $RMIP$, whose optimal objective function value provides a lower bound for $RUFLP$. In Section 3.2, we present a heuristic which is based on a neighborhood search over the location set from the solution to $RMIP$. The exact approach presented in 3.3 is based on successive lower and upper bound improvements for $RUFLP$.

3.1. A Lower Bound for $RUFLP$

Let $q_{[1]} \leq q_{[2]} \leq ... \leq q_{[m-1]} \leq q_{[m]}$ be an ordering of failure probabilities in M. Define

$$Q_{jr} = \begin{cases} (\prod_{t=1}^{r-1} q_{[t]})(1-q_j) & j \in M, \\ \prod_{t=1}^{r-1} q_{[t]} & j = J. \end{cases}$$

Note that Q_{jr} is an *optimistic* version of P_{ijr} $\forall i \in N$ and in the following result, we demonstrate that Q_{jr} is a lower bound for P_{ijr} $\forall i \in N$.

LEMMA 1. *Consider a location set $S \subseteq M$, and for $i \in N$, denote $S[i] = \{k \mid \phi_i \geq c_{ik}, k \in S\}$ and $s_i = |S[i]|$. We have $Q_{jr} \leq P_{ijr}$ $\forall i \in N$, $j \in S \cup \{J\}$, $1 \leq r \leq s_i + 1$.*

Proof: Let $\overline{q}_{[1]} \leq \overline{q}_{[2]} \leq ... \leq \overline{q}_{[s_i-1]} \leq \overline{q}_{[s_i]}$ and $q_{[1]} \leq q_{[2]} \leq ... \leq q_{[m-1]} \leq q_{[m]}$ be an ordering of failure probabilities in S and M, respectively. Then by definition, we have $q_{[t]} \leq \overline{q}_{[t]}$ for $i \in N$, $t = 1, 2, ..., s_i$, which results in $\prod_{t=1}^{r-1} q_{[t]} \leq \prod_{t=1}^{r-1} \overline{q}_{[t]}$ for $i \in N$, $r = 1, 2, .., s_i + 1$. Recall the definition of $i[r, S] \in S$ ($1 \leq r \leq s_i + 1$) as the facility in S that serves customers at $i \in N$ at level r. Then, by definition, we have $\prod_{t=1}^{r-1} \overline{q}_{[t]} \leq \prod_{t=1}^{r-1} q_{i[t,S]}$ for $i \in N$, $r = 1, 2, .., s_i + 1$. Therefore, we conclude $\prod_{t=1}^{r-1} q_{[t]} \leq \prod_{t=1}^{r-1} q_{i[t,S]}$ for $i \in N$, $r = 1, 2, .., s_i + 1$. We note that, for $i \in N$, $r = s_i + 1$ ($j = J$). This is enough to prove $Q_{J(s_i+1)} \leq P_{iJ(s_i+1)}$ and for $i \in N$, $1 \leq r \leq s_i$ ($j = i[r, S]$), we multiply each side of the inequity $\prod_{t=1}^{r-1} q_{[t]} \leq \prod_{t=1}^{r-1} q_{i[t,S]}$ by $(1 - q_j)$, to obtain

$(1-q_j)\prod\limits_{t=1}^{r-1} q_{[t]} \leq (1-q_j)\prod\limits_{t=1}^{r-1} q_{i[t,S]}$. Therefore, $Q_{jr} \leq P_{ijr}\ \forall i \in N, j \in S, 1 \leq r \leq s_i$. Thus, the proof is complete.

By replacing P_{ijr} with fixed failure probabilities $Q_{jr}\ \forall i \in N, j \in M[i]\cup\{J\},\ 1 \leq r \leq m_i+1$ in $RUFLP$ will result the following mixed integer program which we call $RMIP$:

$$(\text{RMIP}) \quad \text{Minimize } Z = \sum_{j\in M} f_j x_j + \sum_{i\in N}\sum_{j\in M[i]\cup\{J\}}\sum_{r=1}^{m_i+1} \lambda_i c_{ij} Q_{jr} y_{ijr}$$

$$\text{s.t.} (4)-(6)$$

$$x_j, y_{ijr} \in \{0,\ 1\}\ \forall i \in N,\ j \in M[i]\cup\{J\},\ 1 \leq r \leq m_i+1.$$

THEOREM 1. *Define* $(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP})$ *and* $Z_{RMIP}(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP})$ *to be the optimal solution and the optimal value of the objective function of RMIP, respectively. Also let* $(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$ *and* $Z_{RUFLP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$ *be the optimal solution and the optimal value of the objective function of RUFLP, respectively.* $Z_{RMIP}(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP})$ *is a lower bound for* $Z_{RUFLP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$, *i.e.*

$$Z_{RMIP}(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP}) \leq Z_{RUFLP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP}). \tag{10}$$

Proof: We note that by construction, $(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$−the optimal solution of $RUFLP$, is a feasible solution for $RMIP$ and from Lemma 1, we obtain $Z_{RMIP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP}) \leq Z_{RUFLP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$. Also since $(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP})$ is the optimal solution of $RMIP$, we have $Z_{RMIP}(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP}) \leq Z_{RMIP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$. Therefore, we conclude $Z_{RMIP}(\mathbf{x}^*_{RMIP}, \mathbf{y}^*_{RMIP}) \leq Z_{RUFLP}(\mathbf{x}^*_{RUFLP}, \mathbf{y}^*_{RUFLP})$, which completes the proof.

We note that $RMIP$ is equivalent to its relaxation without the integrality constraints on $\mathbf{y}$. This will make it easier to solve $RMIP$.

3.2. An Approximate Approach for $RUFLP$

We note that for $R=1$ and $\varphi_i = +\infty\ \forall i \in N$, $RUFLP$ reduces to classical uncapacitated facility location problem ($UFLP$). Since $UFLP$ is NP-hard, $RUFLP$ is NP-hard as well. Thus, it is difficult to obtain good solutions for large size instances of $RUFLP$ within a limited time frame. This fact motivates research on approximate approaches. The heuristic presented below is based on the solution to $RMIP$. Given Theorem 1, the objective value of the solution to $RMIP$ provides a lower bound for $RUFLP$. We note that any feasible

location vector $\mathbf{x}$ including the one produced by solving $RMIP$ generates a feasible solution to $RUFLP$. This is achieved by first defining the assignment vector $\mathbf{y}(\mathbf{x})$ using the assumption that customers are assigned to open facilities level by level in an increasing order of shipping cost. Then, the resulting value of $Z_{RUFLP}(\mathbf{x}, \mathbf{y}(\mathbf{x}))$ provides an upper bound for $RUFLP$.

Denote $S_{\mathbf{x}}$ as the set of facility locations, which are open given location vector $\mathbf{x}$. To find an improved upper bound, the heuristic uses a descent approach in a neighborhood search for $S_{\mathbf{x}}-$the location set produced by solving $RMIP$. For each location set in the neighborhood we find its corresponding sum of shipping and fixed costs assuming customers are assigned to open facilities at $S_{\mathbf{x}}$ level by level in an increasing order of shipping cost. The neighborhood of a location set $S_{\mathbf{x}}$ and the descent approach is defined as follows.

$N_k(S)$, the *distance-k neighborhood* of $S \subseteq M$, is defined as

$$N_k(S) = \{S' \subseteq M : |S - S'| + |S' - S| \leq k\};$$

i.e. S' is in the distance-k neighborhood of S if the number of non-overlapping elements in the two sets does not exceed k.

Once the neighborhood is well defined, the descent algorithm is straightforward: use the solution to $RMIP$ as a starting subset $S_{\mathbf{x}}$; evaluate the change in the value of the objective function for all the subsets in the neighborhood; if an improved subset exists in the neighborhood, move the search to the best vector in the neighborhood. Repeat the process with the new subset until no improved vector exists in the neighborhood. The last subset is the solution. Denote $S_{\overline{\mathbf{x}}}$, the set of facility locations under vector $\overline{\mathbf{x}}$, as the solution subset to the descent approach, and $\mathbf{y}(\overline{\mathbf{x}})$ as the assignment vector where customers are assigned to open facilities at $S_{\overline{\mathbf{x}}}$ level by level in an increasing order of shipping cost. The resulting value of the objective function $Z_{RUFLP}(\overline{\mathbf{x}}, \mathbf{y}(\overline{\mathbf{x}}))$ is our new upper bound and the solution to the descent approach $\overline{\mathbf{x}}$ and $\mathbf{y}(\overline{\mathbf{x}})$ is the solution to our heuristic.

3.3. An Exact Approach for $RUFLP$

The exact approach presented below is based on successive improvements on lower and upper bounds on $RUFLP$. In this approach, we first find initial lower and upper bounds for $RUFLP$ by solving the heuristic proposed in Section 3.2. In the next step, we find an improved lower bound by solving an improved $RMIP$. An improved $RMIP$ is $RMIP$ with

additional cuts which exclude the pre-examined location vectors from the feasible region (at the first step $RMIP$ is solved without any cuts). After finding an improved lower bound, the location set produced by solving the improved $RMIP$ is used as a starting location set in a neighborhood search to find an improved upper bound using the descent approach explained in Section 3.2.

To complete this step, for each starting location set used in the descent approach we introduce a cut to $RMIP$ to exclude all of the feasible location vectors which are in its neighborhood and have already been examined. Denote $S_{\widehat{\mathbf{x}}}$ to be a starting location set in the descent approach, then the following constraint will ensure that all location vectors in the neighborhood of $\widehat{\mathbf{x}}$ (and have already been examined) are infeasible:

$$\sum_{j \in S_{\widehat{\mathbf{x}}}} x_j - \sum_{j \in M - S_{\widehat{\mathbf{x}}}} x_j \leq |S_{\widehat{\mathbf{x}}}| - k - 1. \tag{11}$$

We note that (11) does not make any location vector infeasible unless they are in the neighborhood of $\widehat{\mathbf{x}}$. The addition of this cut to the improved $RMIP$ will help to improve the lower bound in the next steps.

The procedure continues until the gap between the current lower bound and upper bound is within a precision level so that it is evident that the unexamined location sets are unable to improve the current upper bound.

Now consider Problem $RMIP(l)$ as follows:

$$\text{Minimize} Z_{RMIP(l)} = \sum_{j \in M} f_j x_j + \sum_{i \in N} \sum_{j \in M[i] \cup \{J\}} \sum_{r=1}^{m_i+1} \lambda_i c_{ij} P_{jr} y_{ijr}$$

$$\text{s.t.} (4) - (6)$$

$$\sum_{j \in S} x_j - \sum_{j \in M \setminus S} x_j \leq |S| - k - 1, \ \forall S \in A_{RMIP(t)}, \ t = 1, 2, .., l-1, \tag{12}$$

$$x_j \in \{0, \ 1\} \ \forall \ j \in M. \tag{13}$$

Here $A_{RMIP(t)}$ is the set of all starting subsets in the descent approach, which starts the search using the location set found from the solution of $RMIP(t-1)$. Denote $\mathbf{x}^*_{RMIP(t)}$ as the optimal location vector for $RMIP(t)$, and $S_{\mathbf{x}^*_{RMIP(t)}}$ as the set of facility locations, which are open given location vector $\mathbf{x}^*_{RMIP(t)}$. Note that to solve $RMIP(l)$, we need to

solve $RMIP(t)$ for $t = 1, 2, .., l-1$. Note that $RMIP(1)$ does not include constraint (12) and is the original $RMIP$ by definition. Also note that constraints (12) exclude all of the location vectors that have already been examined without excluding unexamined location vectors.

Denote $\mathbf{x}(t)$ to be the location vector solution to the descent approach, which starts its search from location set $S_{\mathbf{x}^*_{RMIP(t)}}$. Denote $\mathbf{y}(\mathbf{x}(t))$ to be the customer assignment vector, where customers are assigned to open facilities in $\mathbf{x}(t)$ level by level in an increasing order of shipping cost. Denote $Z_{RUFLP}(\mathbf{x}(t) , \mathbf{y}(\mathbf{x}(t)))$ as the objective function value of $RUFLP$ given $\mathbf{x}(t) , \mathbf{y}(\mathbf{x}(t))$. Then $UB(t) = \min\{UB(t-1), Z_{RUFLP}(\mathbf{x}(t) , \mathbf{y}(\mathbf{x}(t)))\}$ is the improved upper bound after solving $RMIP(t)$ and performing descent approach which starts the search from location set $S_{\mathbf{x}^*_{RMIP(t)}}$. It is easy to verify that $UB(t)$ is non-increasing in t.

Denote $LB(t) = Z^*_{RMIP(t)}$ as the improved lower bound after solving $RMIP(t)$, where $Z^*_{RMIP(t)}$ is the optimal value of the objective function in $RMIP(t)$. From the formulation of $RMIP(l)$, we have that $LB(t)$ is non-decreasing in t.

Assume a specified precision level $\epsilon \geq 0$, then the search and cut algorithm is described as follows:

The Search and Cut Algorithm

Step 0: Set $l = 1$, $S_{\mathbf{x}^*} = \{\}$, *Lower Bound* $= 0$ and *Upper Bound* $= \infty$.

Step 1: Solve $RMIP(l)$. If a feasible solution exists then find $\mathbf{x}^*_{RMIP(l)}$−the optimal location solution vector in $RMIP(l)$, $S_{\mathbf{x}^*_{RMIP(l)}}$−the set of facility locations under vector $\mathbf{x}^*_{RMIP(l)}$, $\mathbf{y}(\mathbf{x}^*_{RMIP(l)})$−the customer assignment vector, where customers are assigned to open facilities in $\mathbf{x}^*_{RMIP(l)}$ level by level in an increasing order of shipping cost, $Z^*_{RMIP(l)}$−the value of the objective function in $RMIP(l)$, and set *Lower Bound*$(l) = Z^*_{RMIP(l)}$ and proceed to Step 2. Else go to Step 5.

Step 2: If *Lower Bound*$(l) >$ *Lower Bound*, let *Lower Bound* $=$ *Lower Bound*(l). If $\frac{Upper\ Bound - Lower\ Bound}{Lower\ Bound} < \epsilon$, go to Step 5. Otherwise, go to step 3.

Step 3: Perform the descent approach which starts the search from location set $S_{\mathbf{x}^*_{RMIP(l)}}$ and find $A_{RMIP(l)}$−the set of all the starting subsets in the descent approach; $\mathbf{x}(l)$, $\mathbf{y}(\mathbf{x}(l))$−the location and customer assignment solution vectors of the best found solution; $S_{\mathbf{x}(l)}$−the set of facility locations under vector $\mathbf{x}(l)$; and their corresponding objective function value $Z_{RUFLP}(\mathbf{x}(l) , \mathbf{y}(\mathbf{x}(l)))$. Set *Upper Bound*$(l) = Z_{RUFLP}(\mathbf{x}(l) , \mathbf{y}(\mathbf{x}(l)))$.

Step 4: If $Upper\ Bound(l) < Upper\ Bound$, then $Upper\ Bound = Upper\ Bound(l)$, $\mathbf{x}^* = \mathbf{x}(l)$, $\mathbf{y}(\mathbf{x}^*) = \mathbf{y}(\mathbf{x}(l))$ and $S_{\mathbf{x}^*} = S_{\mathbf{x}(l)}$. Set $l = l + 1$ and go to Step 1.

Step 5: Stop. The solution to Search and Cut Algorithm is as follows: The location set is $S_{\mathbf{x}^*}$, location vector is $\mathbf{x}^*$, customer assignment vector is $\mathbf{y}(\mathbf{x}^*)$, and the objective function value $Z^* = Upper\ Bound$.

The algorithm terminates in either Step 1 when all potential location vectors have been examined or Step 5 when the gap between the current upper and lower bound falls below the specified precision level. It is obvious that the algorithm terminates in a finite number of steps. Note that since $Lower\ Bound(l) = Z^*_{RMIP(l)}$, then $Lower\ Bound(l)$ in the algorithm is non-decreasing in l. Since at iteration l, $Upper\ Bound$ represents the value of the best-found feasible solution, and $Lower\ Bound$ is a valid lower bound on all unexamined location vectors, when $\epsilon = 0$ the algorithm is guaranteed to find the optimal solution for $RUFLP$ in finite number of steps.

4.   Computational Results

We tested our exact and heuristic algorithms on two different types of data. The exact algorithm is tested on four data sets with 50, 75, 100, and 150 nodes. These data sets are based on 1990 census data, with each node representing one of the 50, 75, 100 or 150 largest cities in the US. Demands λ_i are set to the city population divided by 10^4, while the fixed cost f_j is set to the median home value in the city. The transportation cost c_{ij} is calculated using $c_{ij} = d_{ij}$, where d_{ij} is the great circle distance between node i and j. We note that the great circle distance between two nodes is the shortest distance of those nodes on the surface of a sphere.

In all four data sets, the set of facilities M is equal to the set of customers N (each demand point is a potential facility site). Penalty cost ϕ_i is set to 10,000 for each customer, and the failure probabilities q_j are calculated using $q_j = \beta + 0.1\alpha e^{-d_j/400}$, where $\beta = 0.01$, and d_j is the great cycle distance (in miles) between node j and New Orleans, LA. For each data set, we fix $\alpha = 1.0$, and vary the maximum assignment level R from 3 to 10; then set at $R = m$. We also fix $R = 4$ and vary α from 1.05 to 1.45 at 0.05 increment. The search-and-cut procedure is executed to a precision of $\epsilon = 0.005$, or up to 3600 seconds in CPU time in the exact algorithm, and executed for a single iteration in the heuristic algorithm. For each "real" test instance, we report the computational times of three different search-and-cut

algorithms, based on distance-k neighborhood, where $k \in \{1, 2, 3\}$. As a comparison to our exact algorithm, we also test the Lagrangian Relaxation algorithm of Cui et al. (2010). To simplify presentation, from now on we will refer the Lagrangian Relaxation algorithm of Cui et al. (2010) as the "LR method".

The heuristic algorithm, which is the approximate approach described in Section 3.2, is based on distance-2 neighborhood ($k = 2$) and in order to show its efficiency, it is tested on a larger problem instances than the ones used for the exact approach. These instances are based on data sets with dense networks with up to 600 nodes given in Beasley (1990). The transportation cost c_{ij} is calculated using $c_{ij} = d_{ij}$, where d_{ij} is the shortest path distance between node i and j. Demands and fixed costs are randomly generated from uniform distributions between 10 and 110, and between 1,000 and 11,000, respectively. Penalty cost is set to be 1,000 for each customer, and failure probabilities are randomly generated from a uniform distribution between 0.01 and 0.11.

Our algorithms are coded in C++ and tested using an Intel Pentium 4 3.20GHz processor with 1.0 GB RAM under Linux and CPLEX package, version 10 was used to solve $RIMP(l)$. Full set of test results are available from the authors. The test results for the exact algorithm are summarized in Tables 1 - 4. The first three columns in Tables 1 - 4 provide the values of the parameter settings. The next four columns provide information about the solution quality: the upper bound and the gap between the lower bound and the upper bound for Search and Cut Algorithm and LR method. The last four columns provide information about computational time of three different Search-and-Cut algorithms, based on distance-k neighborhood $k \in \{1, 2, 3\}$ and LR method. We note that the solution quality of all three Search-and-Cut algorithms remains the same that is why we only report one upper bound and gap for search-and-cut algorithm. The first nine rows in Tables 1 - 4 test the effect of maximum assignment level R, on the solution quality and computational time of the exact methods and the last nine rows test the effect of α, on the solution quality and computational time of the exact methods.

The following conclusions can be drawn from Tables 1-4:

1- The exact approach (Search and Cut Algorithm) was able to solve most of the instances, except a few larger instances.

2- In all cases where the Search and Cut Algorithm could not find the solution within the 3600 seconds limit, the solution found by the algorithm still outperforms the solution found by the method used in LR method.

3- In all cases (except one) where both algorithms found the solution, the Search and Cut Algorithm outperforms the LR method in computational time.

    3-1- For instances with 50 nodes the Search and Cut Algorithm, based on distance-3 neighborhood on average found the solution almost 15 times faster than LR method.

    3-2- For instances with 75 nodes the Search and Cut Algorithm, based on distance-3 neighborhood on average found the solution almost 19 times faster than LR method.

    3-3- For instances with 100 and 150 nodes the LR method could not find any solution, while Search and Cut Algorithm, based on distance-3 neighborhood failed to find the solution in only four instances.

4- The LR method fails to find any solution when problem size is large (100 and 150 nodes instances) or when maximum assignment level R is large.

5- The computational time for the Search and Cut Algorithm seems to have a low sensitivity to the maximum assignment level R, while it is sensitive to α, such that an increase of 25% in α on average will increase the computational time of the Search and Cut Algorithm by almost 400%.

6- The computational time for the LR method is sensitive to α and is very sensitive to the maximum assignment level R.

7- Although the Search and Cut Algorithm, based on distance-3 neighborhood is on average almost 30% faster than the Search and Cut Algorithm, based on distance-2 neighborhood, it fails to outperform when problem size is large (150 nodes instances) and $\alpha = 1$.

8- The computational time for the Search and Cut Algorithm increases exponentially as the number of nodes increases.

From the computational experiments we conclude that the Search and Cut Algorithm outperforms LR method in computational time and solution quality.

The test results for the heuristic algorithm are summarized in Table 5. The first two columns in Table 5 provide the instance number and number of nodes in each instance. The next three columns provide information about the solution quality: the upper and lower bounds and the gap between the lower bound and the upper bound found by the heuristic. The last column provide information about computational time by the heuristic.

As can be seen from Table 5, the heuristic algorithm proposed in Section 3.2 is able to solve extremely large problems of up to 600 nodes in a reasonable amount of time and solution quality.

Table 1   Performance of Exact Algorithms- 50 Nodes

Nodes	R	alpha	SnC UB	SnC Gap	LR UB	LR Gap	SnC-1 Time	SnC-2 Time	SnC-3 Time	LR Time
50	3	1	1,021,060	0.46%	1,020,980	0.48%	14	9	6	23
50	4	1	1,020,540	0.39%	1,020,540	0.50%	26	16	14	54
50	5	1	1,020,520	0.35%	1,020,520	0.41%	28	16	12	84
50	6	1	1,020,520	0.39%	1,020,520	0.43%	38	26	16	180
50	7	1	1,020,520	0.41%	1,020,520	0.45%	40	27	15	273
50	8	1	1,020,520	0.39%	1,020,520	0.49%	44	34	25	626
50	9	1	1,020,520	0.39%	1,020,520	0.44%	43	33	20	908
50	10	1	1,020,520	0.39%	1,020,520	0.48%	47	31	20	1250
50	50	1	1,020,520	0.39%	-	-	56	38	26	-
50	4	1.05	1,021,410	0.50%	1,023,590	0.48%	30	16	14	67
50	4	1.1	1,022,280	0.48%	1,026,650	0.46%	39	18	14	121
50	4	1.15	1,023,160	0.48%	1,029,710	0.47%	56	20	15	136
50	4	1.2	1,024,030	0.49%	1,032,640	0.49%	90	30	22	133
50	4	1.25	1,024,910	0.48%	1,035,590	0.49%	115	36	27	256
50	4	1.3	1,025,790	0.48%	1,038,550	0.49%	174	47	34	208
50	4	1.35	1,026,670	0.49%	1,041,520	0.47%	194	51	35	488
50	4	1.4	1,027,540	0.47%	1,044,510	0.46%	240	61	41	313
50	4	1.45	1,028,370	0.47%	1,047,510	0.50%	290	77	40	474

Table 2   Performance of Exact Algorithms- 75 Nodes

Nodes	R	alpha	SnC UB	SnC Gap	LR UB	LR Gap	SnC-1 Time	SnC-2 Time	SnC-3 Time	LR Time
75	3	1	1,149,130	0.34%	1,149,070	0.48%	27	17	28	195
75	4	1	1,148,590	0.46%	1,148,590	0.49%	38	26	26	273
75	5	1	1,148,580	0.46%	1,148,580	0.47%	52	29	35	382
75	6	1	1,148,580	0.46%	1,148,580	0.42%	95	53	52	540
75	7	1	1,148,580	0.49%	1,148,580	0.41%	101	73	67	708
75	8	1	1,148,580	0.46%	1,148,580	0.39%	124	89	67	2098
75	9	1	1,148,580	0.46%	1,148,580	0.42%	134	81	74	2382
75	10	1	1,148,580	0.46%	1,148,580	0.44%	135	81	76	2444
75	75	1	1,148,580	0.46%	-	-	177	100	95	-
75	4	1.05	1,149,600	0.45%	1,152,670	0.49%	48	31	35	229
75	4	1.1	1,150,600	0.47%	1,156,310	0.50%	60	31	35	254
75	4	1.15	1,151,610	0.48%	1,160,000	0.49%	78	39	40	366
75	4	1.2	1,152,610	0.50%	1,163,720	0.50%	107	52	40	621
75	4	1.25	1,153,620	0.50%	1,167,500	0.50%	138	68	50	824
75	4	1.3	1,154,630	0.44%	1,171,320	0.50%	181	84	56	974
75	4	1.35	1,155,640	0.49%	1,175,190	0.50%	223	99	73	1518
75	4	1.4	1,156,660	0.47%	1,179,110	0.50%	322	128	91	1915
75	4	1.45	1,157,670	0.49%	1,183,090	0.50%	437	147	105	2216

Table 3   Performance of Exact Algorithms- 100 Nodes

Nodes	R	alpha	SnC UB	SnC Gap	LR UB	LR Gap	SnC-1 Time	SnC-2 Time	SnC-3 Time	LR Time
100	3	1	1,254,080	0.48%	1,253,910	0.87%	393	194	255	3612
100	4	1	1,253,010	0.49%	1,253,010	0.76%	1662	488	358	3672
100	5	1	1,252,990	0.49%	1,252,990	0.72%	1968	655	547	3707
100	6	1	1,252,990	0.49%	1,254,560	1.74%	3568	1103	691	3621
100	7	1	1,252,990	0.49%	1,252,990	1.20%	3631	1142	671	3745
100	8	1	1,252,990	0.49%	1,253,460	3.28%	3658	1315	717	3785
100	9	1	1,252,990	0.49%	1,254,310	5.54%	3643	1265	849	3660
100	10	1	1,252,990	0.49%	1,253,460	3.31%	3662	1261	885	3740
100	100	1	1,252,990	0.49%	-	-	3656	1476	1066	-
100	4	1.05	1,254,040	0.48%	1,256,560	0.82%	2668	738	525	3612
100	4	1.1	1,255,060	0.46%	1,260,120	0.85%	3361	854	635	3615
100	4	1.15	1,256,060	0.49%	1,263,280	0.87%	3630	1207	692	3664
100	4	1.2	1,256,920	0.49%	1,267,070	0.94%	3636	1500	863	3618
100	4	1.25	1,257,780	0.49%	1,269,310	1.12%	3632	1848	1233	3632
100	4	1.3	1,258,640	0.48%	1,274,070	1.27%	3635	2301	1329	3749
100	4	1.35	1,259,500	0.47%	1,275,410	1.22%	3633	3106	1607	3633
100	4	1.4	1,260,370	0.46%	1,277,030	1.21%	3628	3633	1759	3613
100	4	1.45	1,261,240	0.50%	1,281,280	1.38%	3635	3641	1903	3610

Table 4   Performance of Exact Algorithms- 150 Nodes

Nodes	R	alpha	SnC UB	SnC Gap	LR UB	LR Gap	SnC-1 Time	SnC-2 Time	SnC-3 Time	LR Time
150	3	1	1,363,780	0.48%	1,371,000	1.48%	451	215	828	3706
150	4	1	1,362,630	0.47%	1,369,790	1.86%	1315	781	1034	4128
150	5	1	1,362,600	0.47%	-	-	2265	1180	1463	-
150	6	1	1,362,600	0.47%	-	-	3661	2292	2041	-
150	7	1	1,362,600	0.47%	-	-	3620	2294	2550	-
150	8	1	1,362,600	0.47%	-	-	3697	2583	2696	-
150	9	1	1,362,600	0.47%	-	-	3608	2722	2729	-
150	10	1	1,362,600	0.47%	-	-	3624	2845	2904	-
150	150	1	-	-	-	-	-	-	-	-
150	4	1.05	1,363,630	0.50%	1,368,280	1.60%	1919	1044	1109	3610
150	4	1.1	1,364,640	0.47%	1,369,450	1.41%	2727	1328	1240	4110
150	4	1.15	1,365,640	0.48%	1,372,880	1.62%	3671	1731	1474	4257
150	4	1.2	1,366,650	0.49%	1,382,480	2.38%	3629	2650	1809	4076
150	4	1.25	1,367,660	0.49%	1,385,280	2.38%	3663	3148	2422	4330
150	4	1.3	1,368,670	0.49%	1,383,290	2.40%	3670	3629	3319	4036
150	4	1.35	1,369,680	0.51%	1,392,260	2.97%	3673	3638	3618	4124
150	4	1.4	1,370,690	0.59%	1,395,780	3.27%	3606	3636	3619	4754
150	4	1.45	1,371,710	0.66%	1,400,550	4.13%	3679	3624	3621	3730

Table 5   Performance of Heuristic Algorithm

Index	Nodes	LB	UB	Gap	CPU Time
1	100	56005	59104	0.055	1
2	100	54119	57506	0.063	0
3	100	57061	60381	0.058	1
4	100	59160	62091	0.050	1
5	100	51212	54949	0.073	1
6	200	79185	82525	0.042	6
7	200	82317	85734	0.042	5
8	200	82458	85961	0.042	5
9	200	79602	82822	0.040	5
10	200	75862	79467	0.048	9
11	300	99306	102391	0.031	14
12	300	97632	101517	0.040	14
13	300	100576	103351	0.028	21
14	300	100142	103767	0.036	16
15	300	97933	101408	0.035	17
16	400	109138	112694	0.033	34
17	400	107013	109946	0.027	231
18	400	115546	118974	0.030	99
19	400	111601	115729	0.037	294
20	400	109719	113261	0.032	177
21	500	117043	121380	0.037	238
22	500	122883	126744	0.031	77
23	500	124377	128559	0.034	1821
24	500	119575	123328	0.031	351
25	500	122169	125422	0.027	90
26	600	130941	137923	0.053	970
27	600	130296	137564	0.056	1602
28	600	126871	129713	0.022	1047
29	600	130393	134152	0.029	318
30	600	138171	141906	0.027	279

5.   Conclusions

In this paper we developed efficient exact and approximate algorithms to solve Reliable Facility Location models with non homogenous failure probabilities. Both approaches are based on finding successive lower and upper bounds. The upper bound is improved using a neighborhood search algorithm performed on the location set solution of $RMIP$, and the lower bound is improved by resolving $RMIP$ with added cuts, which exclude the location sets examined in the neighborhood search from the feasible region. Given the computational results in Section 4, the exact approach is proven to outperform the Lagrangian Relaxation algorithm of Cui et al. (2010) in computational time and solution quality. The approximate algorithm developed also worked very well and was able to solve extremely large problems of up to 600 nodes in a reasonable amount of time and solution quality.

Our findings also bring up new questions for future research. First, we plan to introduce capacity limits into the model, as opposed to the uncapaciated case in this study. Although when facility capacities are given parameters, the complexity of the models and the solution algorithms will not significantly increase, it is also possible to introduce capacity levels for each facility as decision variables, at a certain reservation cost, and let the system determine those levels. Second, only static decision rules are considered in this study, ignoring the duration and the frequency of the facility disruptions. Incorporating these factors into our model will allow us to examine optimal decision rules in a dynamic environment. Finally, we would explore other applications of the reliability model, especially in the field of integrated supply chain design.

References

Aboolian, R., O. Berman, Z. Drezner. 2008. Location and allocation of service units on a congested network. *IIE Transactions* **40** 422–433.

Aboolian, R., O. Berman, Z. Drezner. 2009a. Multiple server center location problem. *Annals of Operations Research* **167**.

Aboolian, R., O. Berman, D. Krass. 2010. Profit maximizing service system design with congestion and elastic demand. Working paper, California State university san marcos.

Aboolian, R., Y. Sun, G.J. Koehler. 2009b. A location-allocation problem for a web services provider in a competitive market. *Eur. J. Oper. Res.* **194** 64–77.

Ball, M.O., F.L. Lin. 1993. A reliability model applied to emergency service vehicle location. *Oper. Res.* **41** 18–36.

Batta, R., J.M. Dolan, N.N. Krishnamurthy. 1989. The maximal expected covering location problem: Revisited. *Transportation Sci.* **23** 277.

Beasley, J.E. 1990. Or-library: distributing test problems by electronic mail. *Journal of the Operational Research Society* **41** 1069–1072.

Berman, O., D. Krass, M.B.C. Menezes. 2007. Facility reliability issues in network p-median problems: Strategic centralization and co-location effects. *Oper. Res.* **55** 332.

Byrka, J., A. Srinivasan, C. Swamy. 2010. Fault tolerant facility location: A randomized dependent lp-rounding algorithm. *IPCO* .

Charikar, M., S. Khuller, D. Mount, G. Narasimhan. 2001. Algorithms for facility location problems with outliers. *Proceedings of Symposium on Discrete Algorithms* .

Chechik, S., D. Peleg. 2010. Robust fault tolerant uncapacitated facility location. *Proceedings of Symposium on Theoretical Aspects of Computer Science* .

Cornuejols, G., M. L. Fisher, G. L. Nemhauser. 1977. Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Sci.* **23** 789–810.

Cui, T., Y. Ouyang, Z.J. Shen. 2010. Reliable facility location under the risk of disruptions. *Oper. Res.* **58** 998–1011.

Daskin, M.S. 1982. Application of an expected covering model to emergency medical service system design. *Decision Anal.* **13** 416–439.

Daskin, M.S. 1983. Maximum expected covering location model: Formulation, properties and heuristic solution. *Transportation Sci.* **17** 48–70.

Daskin, M.S., K. Hogan, C. ReVelle. 1988. Integration of multiple, excess, backup, and expected covering models. *Environment and Planning B* **15** 15–35.

Geoffrion, A.M. 1974. Lagrangean relaxation for integer programming. *Mathematical Programming Study* **2**.

Ghosh, D. 2003. Neighborhood search heuristics for the uncapacitated facility location problem. *Eur. J. Oper. Res.* **150**.

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

Lim, M., M.S. Daskin, S. Chopra, A. Bassamboo. 2009. Managing risks of facility disruptions. Working paper, northwestern university.

Sahin, G., H. Sural. 2007. A review of hierarchical facility location models. *Computers and Operations Research* **34** 2310–2331.

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

Snyder, L.V., M.S. Daskin. 2006. Stochastic p-robust location problems. *IIE Transactions* **38** 971–985.

Swamy, C., D. B. Shmoys. 2003. Fault tolerant facility location. *Proceedings of Symposium on Discrete Algorithms* .

Yan, L., M. Chrobak. 2011. Approximation algorithms for the fault-folerant facility placement problem. *Information Processing Letters* **111** 545–549.

Z.-J.M.Shen, L.R. Zhan, J. Zhang. 2011. The reliable facility location problem: Formulations, heuristics, and approximation algorithms. *INFORMS Journal of Computing* **23** 470–482.