A Constant Factor Approximation Algorithm for the Fault–Tolerant Facility Location Problem [*]

Sudipto Guha [†] Adam Meyerson [‡] Kamesh Munagala [§]

April 3, 2003

Abstract

We consider a generalization of the classical facility location problem, where we require the solution to be *fault–tolerant*. In this generalization, every demand point j must be served by r_j facilities instead of just one. The facilities other than the closest one are "backup" facilities for that demand, and any such facility will be used only if all closer facilities (or the links to them) fail. Hence, for any demand point, we can assign non-increasing weights to the routing costs to farther facilities. The cost of assignment for demand j is the weighted linear combination of the assignment costs to its r_j closest open facilities. We wish to minimize the sum of the cost of opening the facilities and the assignment cost of each demand j. We obtain a factor 4 approximation to this problem through the application of various rounding techniques to the linear relaxation of an integer program formulation. We further improve the approximation ratio to 3.16 using randomization and to 2.41 using greedy local-search type techniques.

1 Introduction

The facility location problem has been used as a model in network design and location theory: placement of routers or caches [21, 11], plants or warehouses [17, 1, 26], agglomeration of traffic or data [2, 12], among others (refer to [9] for a more exhaustive list). The problem, given a set of demand and facility locations, tries to minimize the sum of the cost of building facilities at a subset of facility locations and the cost of assigning every demand to a built facility. It models the tradeoff of developing resources (facilities) and the utility (reduction in assignment cost) accruing from such. In several applications, caching on a network, for example, fault tolerance is also a facet. The placement of caches should be resistant to failures of nodes and links. The facility location problem does not provide any guarantee about the second closest facility to any node. In a fault–tolerant situation, the cost of a location that requires a "backup" would be a combination of the costs of assigning a demand location to the two facilities. A natural choice could be a weighted linear combination.

[*] A preliminary version of this paper was published in the proceedings of twelfth SIAM symposium on Discrete Algorithms, 2001.

[†] Department of Computer Information Science, University of Pennsylvania. This work was done while the author was at the Department of Computer Science, Stanford University, CA 94305, and his research was supported by an IBM Research Fellowship and NSF Grant IIS-9811904. Email: sudipto@cis.upenn.edu.

[‡] Currently at Department of Computer Science, Carnegie Mellon University and would be at Department of Computer Science, UCLA, starting Fall 2003. This work was done while the author was at Department of Computer Science, Stanford University, Palo Alto, CA 94305 an his research was supported by ARO DAAG-55-97-1-0221. Email: awm@cs.stanford.edu

[§] Department of Computer Science, Stanford University CA 94305. Supported by ONR N00014-98-1-0589. Email: kamesh@cs.stanford.edu.

In this paper we consider the problem of fault–tolerant facility location in which every location j specifies to be assigned to r_j facilities. The cost of assignment of this location is a weighted combination of these r_j assignments. Recently Jain and Vazirani [15] provided a primal dual approximation whose approximation ratio is logarithmic in the largest requirement r_j. In their algorithm, even for all requirements being 1, the approximation is at least 3. In contrast the fault–tolerant variant of the k–center problem, which is closely related to the facility location problem, has constant factor approximation algorithms [6, 18, 20, 25]. We resolve this issue by providing a constant factor approximation for the fault–tolerant facility location problem. Our result improves on [15] even if the maximum requirement is 1.

Our algorithm is based on rounding the relaxation of an integer linear program. We use filtering in a fashion similar to [22, 26], however we combine it with scaling and uncrossing steps. These steps allow us to ensure that while we are considering a filtered neighborhood, if a demand point is assigned, we will round in such a way that the entire assignment can be rearranged to maintain feasibility.

Finally, we demonstrate another facet in which fault tolerance does not impact approximability of facility location. This is the idea of local improvement heuristics. We use greedy local improvement similar to [10] to construct a solution of integrality gap 2.408. The core of the similarity is that once a set of facilities are fixed, the assignment costs are also determined. However, the similarity does not seem to extend to allow us to delete facilities as in the combinatorial facility location algorithms in [19, 4]. Very recently [27] obtain stronger results using other paradigms of approximating uncapacitated facility location to the fault–tolerant case.

2 Problem Statement

In the fault–tolerant facility location problem, we are given a finite metric $G = (V, E)$ with a distance function c, a set of possible facility locations $\mathcal{F} \subseteq V$, and a set of demand points $\mathcal{D} \subseteq V$. The subscripts i, j will be used to denote facilities and demand points respectively throughout this paper. The cost of opening a facility at location i is f_i. Every demand j must be connected to r_j open facilities.

For demand j, let the weights corresponding to assigning j to the r_j facilities be $w_j^{(1)} \geq w_j^{(2)} \geq \cdots \geq w_j^{(r_j)}$. Naturally this would ensure that the open facilities to which j is connected to, would be ordered according to the (increasing) distance from j. The goal is to optimize the sum of the cost of open facilities and the weighted sum of the routing costs of each demand to the closest open facilities. We assume *unit* demands. The algorithm remains exactly the same for general demands, since the demands can be incorporated in the weights $w^{(r)}$.

This problem can be formulated as an integer program. Here, y_i denotes whether facility i is open, and $x_{ij}^{(r)}$ denotes that demand j is assigned to facility i and facility i is the r^{th} closest open facility to j. The distance between i and j is c_{ij}.

$$\text{Minimize} \sum_i \sum_j \sum_r c_{ij} w_j^{(r)} x_{ij}^{(r)} + \sum_i f_i y_i$$

$$\begin{array}{rcll} \sum_i x_{ij}^{(r)} & \geq & 1 & \forall j, r \\ \sum_r x_{ij}^{(r)} & \leq & y_i & \forall i, j \\ y_i & \leq & 1 & \forall i \\ x_{ij}^{(r)}, y_i & \in & \{0, 1\} & \forall i, j, r \end{array}$$

The relaxation will involve relaxing the last constraints to $0 \leq x_{ij}^{(r)}, y_i \leq 1$. The upper bound is only relevant for y_i and ensures that more than one facility is not built at a location.

Define C^* to be the optimal fractional assignment cost and F^* to be the optimal fractional facility cost. That is

$$\sum_i \sum_j \sum_r c_{ij} w_j^{(r)} x_{ij}^{(r)} = C^* \text{ and } \sum_i f_i y_i = F^*$$

where (x, y) denotes the optimal fractional solution of the above linear programming relaxation.

2.1 Previous Results

Classical facility location is MAX-SNP hard [10], and several constant factor approximations [26, 4, 16] are known. Since the problem we study is a generalization of this problem, the hardness results carry over. Many variants of the facility location problem have been studied. The more well known ones include capacitated facility location [26, 16], multi-level facility location [1, 11], k-center [13], and k-median [22, 5, 16, 4]. All these problems have constant factor approximation algorithms.

Jain and Vazirani [15] defined the fault–tolerant facility location problem. They assign equal weights to all the facilities a demand is connected to. They present a $O(\log \max_j r_j)$ primal-dual approximation algorithm for this problem. Constant factor approximation algorithms are known for the fault–tolerant k–center problem [6, 18, 20, 25], where each demand point j is required to have r_j centers within a fixed distance L from it.

2.2 Subsequent Work

There has been a lot of interesting work on the facility location problem and its variant the k-median [24, 3, 23, 14].

In [27] a factor 2.07 approximation algorithm was provided for the problem considered here starting from the algorithms of [7, 8]. They further improve the factor to 1.52 for the uniform requirement case (r_j values are 0 or r for some fixed r) adapting techniques of [14].

3 Constructing a Structured Fractional Solution

The linear relaxation of the above-mentioned integer program gives us a fractional solution. We will convert the solution (x, y) to a solution $(\bar{x}, y)$ such that the cost of the new solution does not increase, and the new solution satisfies certain useful properties.

We will treat a demand point j as having r_j copies under the constraint that no two copies of any demand point are assigned to the same facility. In the fractional setting this reduces to the condition $\sum_r x_{ij}^{(r)} \leq y_i \leq 1$. The converted solution will ensure that the set of facilities to which a copy $j^{(r_1)}$ is fractionally assigned to are closer to j than any facility to which the copy $j^{(r_2)}$ is assigned to fractionally for $r_1 < r_2$.

For every demand point j, we reassign it to facilities, fractionally, as follows. Order the facilities in non-decreasing distance from j, breaking ties arbitrarily. The ordering for a specific demand point j is fixed throughout the rest of the algorithm. The first demand copy $j^{(1)}$, is assigned to the initial set of facilities that sum up to 1 fractionally. The last facility i in this set can be incompletely assigned, i.e. $\bar{x}_{ij}^{(1)} < y_i$. For the second copy, we start from this facility i, setting $\bar{x}_{ij}^{(2)} = y_i - \bar{x}_{ij}^{(1)}$. After that we again pick up one unit of facility fractionally, so that $\sum_i \bar{x}_{ij}^{(2)} = 1$. We repeat this process for all the copies of the demand point.

Definition 3.1 *Define $\mathcal{C}_j^{(r)} = \sum_i \bar{x}_{ij}^{(r)} c_{ij}$. Define $\mathcal{C}_j^{(r)}(\beta)$ to be the distance at which the r^{th} copy of the demand point j picks up at least β fraction of a facility; therefore we have, $\int_0^1 \mathcal{C}_j^{(r)}(\beta) d\beta = \mathcal{C}_j^{(r)}$.*

The following are true by construction:

Proposition 3.1 *The cost of the solution does not increase; $\sum_{j,r} w_j^{(r)} \mathcal{C}_j^{(r)} = C^*$.*

Proposition 3.2 *For any facility i and demand j, there exist at most two values of r such that $\bar{x}_{ij}^{(r)} > 0$. Further, if two such values exist they must be consecutive.*

Once the (fractional) facilities are fixed, it is simple to see that the above reassignment is (one of) the best possible. Intuitively, the copies of the demand j with larger weight w_j^r (and thus smaller r) go to the closer open facilities.

4 The Algorithm

The algorithm rounds the fractional solution in two phases. The algorithm uses the filtering technique of Lin and Vitter [22] combined with reassignment of the fractional demands, such that each copy of the demand goes to a different facility. As in the previous section, we treat the different copies of a demand as separate, and denote the r^{th} copy of demand j by $j^{(r)}$. Fix $\alpha \in (0, 1)$, to be determined later.

4.1 Phase 1: Filtering and Scaling

In this section we will modify the fractional solution $(\bar{x}, y)$ to create a new solution $(\hat{x}, \hat{y})$, which we will round in the next phase. This phase uses the filtering technique of [22].

Let us fix a demand point j. We will perform the following operations for the copies $j^{(r)}$ in increasing order of $r = 1, 2 \ldots$. For every demand $j^{(r)}$, we consider the facilities to which it is fractionally assigned in increasing order of distance (the same ordering used in the previous section).

Let i be the first facility in the ordering of $j^{(r)}$ (therefore $x_{ij}^{(r)} > 0$) such that

$$\sum_{i' \,:\, c_{i'j} < c_{ji},\, x_{i'j}^{(r)} > 0} \hat{x}_{i'j}^{(r)} \geq 1 - \alpha$$

In other words, i is the nearest facility to j such that within the distance c_{ij}, $j^{(r)}$ picks up $1 - \alpha$ fraction of a facility.

For all i' appearing before i in our ordering, we set $\hat{x}_{i'j}^{(r)} = \bar{x}_{i'j}^{(r)}$. We set $\hat{x}_{ij}^{(r)}$ so that the total assignment of $j^{(r)}$ is *exactly* $1 - \alpha$. For all i' appearing after i in the ordering, we set $\hat{x}_{i'j}^{(r)} = 0$.

We scale the $\hat{x}_{ij}^{(r)}$ by $\frac{1}{1-\alpha}$ so that $\sum_i \hat{x}_{ij}^{(r)} = 1$ for all $j^{(r)}$. Subsequently for all i we set $\hat{y}_i = \min\left\{\frac{y_i}{1-\alpha}, 1\right\}$.

Lemma 4.1 [22] *If $\hat{x}_{ij}^{(r)} > 0$, then $c_{ij} \leq \frac{1}{\alpha}\mathcal{C}_j^{(r)}$.*

We first show that $(\hat{x}, \hat{y})$ is feasible. For this, it is enough to show the following lemma:

Lemma 4.2 *For all i, j, we have $\sum_r \hat{x}_{ij}^{(r)} \leq \hat{y}_i$.*

Proof: Before filtering, by Proposition 3.2, we knew that at most two copies of a demand went to any one facility. Suppose we are considering facility i and demand j. If exactly one copy, say r is assigned to i, the inequality trivially holds, as $\hat{x}_{ij}^{(r)} \leq \hat{y}_i$.

We therefore assume that two copies of j are assigned to i. Let $j^{(r)}$ and $j^{(r+1)}$ be assigned to i. Note that by the construction in Section 3, i is the furthest assigned facility to $j^{(r)}$ and the closest to $j^{(r+1)}$.

The interesting case is $y_i \geq 1-\alpha$, otherwise $\sum_r \hat{x}_{ij}^{(r)} \leq y_i$ was true before scaling, and the lemma follows as we scale both the left and right hand sides by the same amount.

Let us look at the $\hat{x}_{ij}^{(r)}$ values before scaling (but after filtering). Therefore we need to show $\sum_r \hat{x}_{ij}^{(r)} \leq 1-\alpha$, then, scaling could not have increased this value beyond 1. When we were considering $j^{(r)}$ for filtering, we must have set $\hat{x}_{ij}^{(r)} = \max(0, \bar{x}_{ij}^{(r)} - \alpha)$, as i is the furthest assigned facility to $j^{(r)}$. We now consider two cases:

Case 1: $\hat{x}_{ij}^{(r)} = 0$. Then, $\hat{x}_{ij}^{(r+1)} \leq 1-\alpha$ because of filtering on $j^{(r+1)}$.

Case 2: $\hat{x}_{ij}^{(r)} = \bar{x}_{ij}^{(r)} - \alpha$. This implies $\hat{x}_{ij}^{(r)} + \hat{x}_{ij}^{(r+1)} = \bar{x}_{ij}^{(r)} + \bar{x}_{ij}^{(r+1)} - \alpha \leq 1-\alpha$, as $\bar{x}_{ij}^{(r)} + \bar{x}_{ij}^{(r+1)} \leq y_i \leq 1$.

This completes the proof. ■

Lemma 4.3 *Let $r_1 < r_2$. For any demand j, the furthest (from j) facility to which $j^{(r_1)}$ is assigned to (fractionally) is at a distance no greater than the closest (from j) facility to which $j^{(r_2)}$ is assigned to (fractionally) in the filtered and scaled solution.*

Proof: The rearrangement from Section 3 guarantees this on the un-filtered solution; filtering does not change the ordering of the edges. ■

4.2 Phase 2: Rounding

In this phase we will round the fractional solution $(\hat{x}, \hat{y})$ as produced in the previous phase. We will perform a rounding similar to [22, 26], and preserve $\sum_r \hat{x}_{ij}^{(r)} \leq \hat{y}_i$ as an invariant.

The scheme from [26] cannot be applied directly, since the distinct copies of a demand need to be assigned to distinct facilities. The way we ensure this is to pick just enough fractions of facilities to merge so that one copy of the demand can be completely satisfied. We then perform uncrossing of neighborhoods so that the other copies of that demand are assigned to facilities outside the set of facilities we picked for rounding.

- **Step A – Ordering the Demands:** Arrange all copies of all demand points in increasing order of the distance to the farthest fractional facility serving it. We will process the copies in this order, and repeatedly apply Steps B – E. Note that copies of j will be picked in increasing order.

- **Step B – Choosing a Facility:** Assume we are considering $j^{(r)}$, the r^{th} copy of the demand point j. Let the set of facilities serving it be $P_j^{(r)}$.

  We will build a facility at the cheapest facility i in $P_j^{(r)}$.

- **Step C – Merging Facilities:** We now specify a set $\hat{P}$ of (fractional) facilities which will be closed down in exchange for the facility to be opened at i. In other words, we can view this set as a set of fractional facilities to be merged into i. The set will have the property that $\sum_{i' \in \hat{P}} \hat{y}_{i'} = 1$.

  1. We select facilities i' with $\hat{x}_{i'j}^{(r)} > 0$ starting with i (order does not matter) until the total fraction by which selected facilities are open is at least 1. Let $Y = \sum_{i'} \hat{y}_{i'} \geq 1$ be the total fraction by which these facilities are open.
  2. If $Y > 1$ we will have to use the last selected facility, say i'', partially. Make two copies, i_1 and i_2, of facility i''. Set $\hat{y}_{i_2} = Y - 1$, and $\hat{y}_{i_1} = \hat{y}_{i''} - \hat{y}_{i_2}$. For any other demand $j'^{(r')}$ the assignment $\hat{x}_{i''j'}^{(r')}$ is distributed arbitrarily between the two facility copies i_1 and i_2; maintaining $\sum_r \hat{x}_{i'j}^{(r)} \leq \hat{y}_{i'}$ for both $i' = i_1$ and $i' = i_2$. The facility (copy) i_1 is selected and i_2 is not. Denote the set of picked facilities by $\hat{P}$.[1]

  We open a facility completely at i, and close the rest of the facilities in the set $\hat{P}$.

- **Step D – Assignment of Demands:** For any demand j' (inclusive of j), consider its copies $r_1, r_2, \ldots, r_k$ served at least fractionally by $\hat{P}$. If $\hat{P}$ serves any copy of j' fractionally, we assign the smallest numbered copy (r_1) of j' to be completely served by i. Note that the assignment distance for $j'^{(r_1)}$ has at most tripled as compared to $C_{j'}^{(r)}(1-\alpha)$, see proof of Lemma 4.6.

- **Step E – Uncrossing Neighborhoods:** We now reassign the remaining copies of j' (i.e. $j'^{(r_2)}$, ..., $j'^{(r_k)}$) completely to facilities outside the set $\hat{P}$ by performing an uncrossing step.

  For j', we compute $X_{j'}^{(1)} = \sum_{i' \in \hat{P}} \hat{x}_{i'j'}^{(r_1)}$, and $X_{j'}^{(2)}, \ldots, X_{j'}^{(k)}$ likewise. These quantities denote the fractions to which the copies of j' are assigned to the facilities in $\hat{P}$. define $Y_{j'}^{(1)} = \sum_{i' \notin \hat{P}} \hat{x}_{i'j'}^{(r_1)} = 1 - X_{j'}^{(1)}$, and similarly $Y_{j'}^{(2)}, \ldots, Y_{j'}^{(k)}$. These quantities denote the fractions by which the copies of j' are assigned to facilities outside the set $\hat{P}$, respectively. The following is achieved by the construction:

  Proposition 4.4 *For any j' which is fractionally assigned to the facilities in set $\hat{P}$, we have:*

  $X_{j'}^{(t)} + Y_{j'}^{(t)} = 1$ *for all* $1 \leq t \leq k$, *and*

  $\sum_t X_{j'}^{(t)} \leq \sum_{i' \in \hat{P}} \hat{y}_{i'} = 1$.

  We have assigned the copy $j'^{(r_1)}$ to i. But in this process it may be that $X_{j'}^{(r')} > 0$ that is $\hat{P}$ serves some other copy $j'^{(r')}$ of j'. If we use the fractional facility of $\hat{P}$ (which amounts to 1) then we need to ensure that the copy $j'^{(r')}$ gets assigned (fractionally) to facilities outside $\hat{P}$; and the fraction is $X_{j'}^{(r')}$. Notice that in this case from Proposition 4.4,

$$X_{j'}^{(r')} + X_{j'}^{(1)} \leq 1 = X_{j'}^{(1)} + Y_{j'}^{(1)}$$

[1] We have $i_1 \in \hat{P}$ and $i_2 \notin \hat{P}$. Note that since a facility is being built at i, we can build a facility later at $i'' \neq i$.

We consider the fraction $Y_{j'}^{(1)}$ by which the copy $j'^{(r_1)}$ was assigned to facilities not in $\hat{P}$, and re-assign this to the other copies of j' which were originally assigned to the set $\hat{P}$ as follows: Consider the fraction by which $j'^{(r_1)}$ was assigned to the facility closest to j but not in $\hat{P}$. We assign this fraction to $j'^{(r_2)}$ until either $j'^{(r_2)}$ is completely satisfied, or we have assigned the fraction completely. In the former case, we move to $j'^{(r_3)}$; in the latter case, we consider the next closest facility not in $\hat{P}$ that was previously connected to $j'^{(r_1)}$, and repeat.

During uncrossing, we maintain the invariants $\sum_{i'} \hat{x}_{i'j'}^{(r_t)} = 1$ and $\sum_t \hat{x}_{i'j'}^{(r_t)} \leq \hat{y}_{i'}$ for all $1 \leq t \leq k$.

Lemma 4.5 *For any j', we can always re-assign the copies $r_2, r_3, \ldots, r_k$ completely outside set $\hat{P}$ by the uncrossing step.*

Proof: Note that since we use the assignments of r_1 (and r_1 was the smallest numbered copy), by Lemma 4.3, the cost of the solution can only reduce.

Consider the total fraction by which demand j' is assigned outside the set $\hat{P}$. This was originally $\sum_t Y_{j'}^{(t)}$. We have to show that after reassignment, the final total fraction is no more than this. We remove fraction $Y_{j'}^{(1)}$ (because we assign $j'^{(r_1)}$ completely in set $\hat{P}$) and add $\sum_t X_{j'}^{(t)} - X_{j'}^{(1)}$ because of the uncrossing. Therefore, the final fraction is:

$$\sum_t Y_{j'}^{(t)} + \sum_t X_{j'}^{(t)} - (Y_{j'}^{(1)} + X_{j'}^{(1)})$$

Invoking Proposition 4.4, the final fraction is clearly at most $\sum_t Y_{j'}^{(t)}$. This means that the re-assignment is always possible. ■

At the end of one iteration of Steps B – E, we have opened facility i completely. For every demand fractionally assigned to the set $\hat{P}$, the smallest assigned copy is completely assigned to i. Every other copy is fractionally re-assigned completely outside set $\hat{P}$. We drop the set $\hat{P}$ and the copies $j'^{(r_1)}$ from further consideration.

Using arguments similar to [22] and [26], it follows:

Lemma 4.6 *The r'th copy of a demand point j is assigned within a distance $\frac{3}{\alpha}\mathcal{C}_j^{(r)}$, thus the service cost is at most $\frac{3}{\alpha}C^*$. The facility cost of the above solution is at most $\frac{1}{1-\alpha}F^*$.*

Proof: Since $\sum_{i \in \hat{P}} y_i = 1$, and we charge this to the cheapest facility, the facility cost cannot go up in this step. But since we scaled the $\hat{y}_i$ in Phase 1, our cost could go up by $\frac{1}{1-\alpha}$. Since the distances form a metric, and we are using the demand with the smallest $\hat{\mathcal{C}}_j^{(r)}(1-\alpha)$, the distance cannot have more than tripled. Note that since we use the assignments of r_1 (and r_1 was the smallest numbered copy), by Lemma 4.3, the new assignments introduced in uncrossing can only decrease. This, combined with the distance bound from Lemma 4.1 completes the proof. ■

Setting $\alpha = \frac{3}{4}$ we have:

Theorem 4.1 *Fault–tolerant facility location has a factor 4 approximation in polytime.*

The rounding phase requires $O(|\mathcal{D}|^2|\mathcal{F}|^2) = O(n^4)$ steps.

4.3 A Tighter Analysis

Recall from Definition 3.1 that $\mathcal{C}_j^{(r)}(\beta)$ is the distance from j such that at least β fraction of the demand for copy r of j is satisfied. Thus $\mathcal{C}_j^{(r)} = \mathcal{C}_j^{(r)}(1)$.

For a particular choice of α, the cost of the solution, $S(\alpha)$, is:

$$S(\alpha) \leq \frac{1}{1-\alpha}F^* + 3\sum_{j,r} w_j^{(r)}\mathcal{C}_j^{(r)}(1-\alpha) \tag{1}$$

We present the analysis of the algorithm if α were chosen at random from the interval $(0, 1-x)$. Since there are at most $|\mathcal{D}||\mathcal{F}| = O(n^2)$ values of α for which the rounding can be different; the algorithm can be derandomized. Following analysis from [26], the average cost evaluates to:

$$\frac{3}{1-x}C^* + \frac{\ln\frac{1}{x}}{1-x}F^*$$

The above expression is minimized for $x = e^{-3}$, resulting in the following:

Theorem 4.2 *Fault–tolerant facility location has a* 3.16 *approximation algorithm in polytime.*

4.4 Facility Location Revisited

In this section, we will show how to improve the approximation factor, and demonstrate the similarities between uncapacitated facility location and the fault–tolerant version. Consider the heuristic that repeatedly chooses a facility to add while the total cost reduces. The heuristic is known as the 'add' heuristic in the facility location literature. [10, 19, 4] analyze the heuristic (with some variations specific to the analysis) from a standpoint of approximation algorithms. We follow the analysis of [10].

Define the Gain(i) of a facility i to be the decrease in *total cost* (decrease in assignment cost minus the facility cost of i) of the solution on addition of facility i to the solution. The facility with the best gain ratio is the facility i_m with the ratio $\max_i$ Gain$(i)/f_i$. If the Gain(i_m) is positive, the heuristic adds i_m and repeats; and stops otherwise.

The computation of the assignment cost is easy following the observation that once the set of facilities are fixed, every demand point chooses the facilities serving it in increasing order of distance. The improvement of the solution depends on the quality of gain we can guarantee at every step; the following lemma can be proved:

Lemma 4.7 *If the current costs of facility and assignment are F and C, respectively, and there exists a fractional solution with costs F^* and C^* satisfying $C \geq F^* + C^*$, then there exists a node i with ratio Gain$(i)/f_i \geq (C - F^* - C^*)/F^*$.*

Proof: Consider the fractional solution $(\bar{x}, \bar{y})$ with facility and assignment costs F^* and C^*. Without loss of generality we can assume that (a copy of) a demand point either uses a fractional facility completely or not at all. This can be achieved by replicating a facility as in Step C in the previous subsection.

Now consider a facility i in the fractional solution which is open with fraction $\bar{y}_i$. In the fractional solution consider the set $calS_i = \{j^{(r)}\}$ of the copies of the demand point assigned to i. By the assumption that a copy of a demand uses a (fractional) facility completely or not at all in the fractional solution – we are guaranteed that the set $calS_i$ does not contain two copies $j^{(r)}$ and $j^{(r')}$ of the same demand point j.

Consider (for the purpose of proof only) adding this facility *integrally* to our current solution and change the assignment of the demand copies $calS_i$ to the newly added facility i. Define Gain$'(i)$ as

$$\text{Gain}'(i) = -f_i + \sum_{j^{(r)} \in calS_i} w_j^{(r)} \left(\text{“current assignment distance of } j^{(r)}\text{”} - c_{ij} \right)$$

We can interpret Gain$'(i)$ easily if i were not already open, i.e $i \notin F$. In this case Gain$'(i)$ is the change if the facility i were opened and all demand copies $calS_i$ served by i in the fractional solution $(\bar{x}, \bar{y})$ were to be assigned integrally to i. If i is in the current set of open facilities F, then the cost of the facility is paid again.

Observe that since the facility i cannot serve two copies of the same demand point j, the solution as interpreted above is a feasible solution.

Notice at this step we may be making possibly suboptimal assignments to this facility i; and thus Gain$'(i) \leq$ Gain(i). This is true since once we fix the set of facilities, we compute the best possible assignment solution and Gain(i) is the maximum over all possible reassignments.

Now consider the last equation multiplied with $\bar{y}_i$ and summed up as i ranges over the facilities in the fractional solution. We get

$$\begin{aligned}\sum_i \bar{y}_i \text{Gain}'(i) &= -\sum_i \bar{y}_i f_i - \sum_i \sum_{j^{(r)} \in calS_i} \bar{y}_i w_j^{(r)} c_{ij} \\ &\quad + \sum_i \sum_{j^{(r)} \in calS_i} w_j^{(r)} \bar{y}_i \text{ current assignment distance of } j^{(r)}\end{aligned}$$

The first term sums to $-F^*$ which is the fractional facility cost. The second term is the cost of fractional assignment of all the demand copies and is $-C^*$. For the last term the sum can be rewritten, switching the order of summations, as

$$\sum_{j^{(r)}} w_j^{(r)} \text{ “current assignment distance of } j^{(r)}\text{”} \sum_{i : j^{(r)} \in calS_i} \bar{y}_i$$

In the fractional solution the sum $\sum_{i:j^{(r)} \in calS_i} \bar{y}_i$ has to be 1, or in other words the demand copy is fractionally assigned to a total of 1. Thus the term sums to exactly the current assignment cost C. Thus

$$\begin{aligned}C - C^* - F^* = \sum_i \bar{y}_i \text{Gain}'(i) &= \sum_i \bar{y}_i f_i \frac{\text{Gain}'(i)}{f_i} \\ &\leq \sum_i \bar{y}_i f_i \max\{0, \max_i \frac{\text{Gain}(i)}{f_i}\}\end{aligned}$$

Now if $C > C^* + F^*$ then there must exist one i such that Gain$'(i)$ is positive. Morover in that case if i_m is the facility with maximum positive Gain$'(i)/f_i$ then

$$C - C^* - F^* \leq F^* \frac{\text{Gain}'(i_m)}{f_{i_m}}$$

Since $\text{Gain}(i) \geq \text{Gain}'(i)$ for all i, the lemma is proved. ■

The above argument is exactly the same for the original facility location problem, as proved in [4]. This is because in the fractional solution the assumption that a fractional facility serves a demand point completely or not at all allows us to treat the copies of a demand point as separate demand points.

Thus the complete algorithm is as follows: We solve the linear program and perform the rounding as described in Section 4.2. We then repeatedly choose a facility to add to our solution which gives us the best gain. We stop when adding no other facility gives us any gain. The post-processing after the rounding phase takes time $O(|\mathcal{D}||\mathcal{F}|^2) = O(n^3)$ since we add at most $\mathcal{F}$ more facilities.

The next lemma follows from the above lemma and the analysis in [10], or a simpler one in [4].

Lemma 4.8 *If the initial cost of facilities is F and of assignment cost is $C > F^* + C^*$, then at the end of the heuristic the cost is $F + F^* + C^* + F^* \ln \frac{C - C^*}{F^*}$, if there exists a (possibly fractional) solution of facility cost F^* and assignment cost C^*.*

The details of the proof are exactly the same as in [10, 4], and we omit it.

Theorem 4.3 *The Fault–Tolerant facility location problem has a 2.408 approximation algorithm in poly-time.*

It is interesting to note that the combinatorial algorithms proposed in [19, 4] do not extend since deletion of a node cannot be allowed – since it may render a solution infeasible. Both these algorithms employ a pure delete operation - where the number of facilities decreases. A combinatorial algorithm employing only addition of facilities and pure exchanges would very likely extend to the fault–tolerant case.

Acknowledgments

We would like to thank the referees for their meticulous review which improved the presentation significantly.

References

[1] K. Aardal, F. Chudak, and D.B. Shmoys. A 3-approximation algorithm for the k-level uncapacitated facility location problem. *Information Processing Letters*, 72:161–167, 1999.

[2] M. Andrews and L. Zhang. The access network design problem. *Proceedings of 39th IEEE FOCS*, pages 40–59, 1998.

[3] V. Arya, N. Garg, R. Khandekar, K. Munagala, and V. Pandit. Local search heuristic for k-median and facility location problems. *Proceedings of* 33^{th} *ACM STOC*, pages 21–29, 2001.

[4] M. Charikar and S. Guha. Improved combinatorial algorithms for facility location and k-median problems. *Proceedings of* 40^{th} *IEEE FOCS*, pages 378–388, 1999.

[5] M. Charikar, S. Guha, É. Tardos, and D. Shmoys. A constant factor approximation algorithm for the k-median problem. *To appear in JCSS,* a preliminary version appeared in *Proceedings of 31st ACM STOC*, pages 1–10, 1999.

[6] S. Chaudhury, N. Garg, and R. Ravi. The p-neighbor k-center problem. *Information Processing Letters*, 65(3):131–134, 1998.

[7] F. Chudak and D. Shmoys. Improved approximation algorithms for the uncapacitated facility location problem. *Unpublished manuscript*, 1998.

[8] F. A. Chudak. Improved approximation algorithms for uncapacitated facility location. *Proceedings of Integer Programming and Combinatorial Optimization*, LNCS 1412:180–194, 1998.

[9] G. Cornuéjols, G.L. Nemhauser, and L.A. Wosley. The uncapacitated facility location problem. In P. Mirchandani and R. Francis, editors, *Discrete Location Theory*, pages 119–171. John Wiley and Sons, Inc., New York, 1990.

[10] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31(1):228–248, 1999.

[11] S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. *Proceedings of* 41^{st} *IEEE FOCS*, pages 603–612, 2000.

[12] S. Guha, A. Meyerson, and K. Munagala. A constant factor approximation for the single sink edge installation problems. *Proceedings of* 33^{rd} *ACM STOC*, pages 383–388, 2001.

[13] D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Math of Operations Research*, 10(2):180–184, 1985.

[14] K. Jain, M. Mahdian, and A. Saberi. A new greedy approach for facility location problems. *Proceedings of* 34^{th} *ACM STOC*, pages 731–740, 2002.

[15] K. Jain and V. Vazirani. An approximation algorithm for the fault tolerant metric facility location problem. *Proceedings of APPROX*, LNCS 1913:177–183, 2000.

[16] K. Jain and V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of ACM*, 48(2):274–296, 2001.

[17] L. Kaufman, M. vanden Eede, and P. Hansen. A plant and warehouse location problem. *Operations Research Quarterly*, 28:547–557, 1977.

[18] S. Khuller, R. Pless, and Y. Sussmann. Fault tolerant k-center problems. *Theoretical Computer Science*, 242(1–2):237–245, 2000.

[19] M. R. Korupolu, G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. *Journal of Algorithms*, 37(1):146–188, 2000.

[20] S. Krumke. On a generalization of the p center problem. *Information Processing Letters*, 56:67–71, 1995.

[21] B. Li, M. Golin, G. Italiano, X. Deng, and K. Sohraby. On the optimal placement of web proxies in the internet. *Proceedings of INFOCOM*, pages 1280–1290, 1999.

[22] J.-H. Lin and J. S. Vitter. ϵ-approximations with minimum packing constraint violations. *Proceedings of* 24^{th} *ACM STOC*, pages 771–781, 1992.

[23] M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. A greedy facility location algorithm analyzed using dual fitting. *Proceedings of RANDOM-APPROX*, LNCS 2129:127–137, 2001.

[24] R. Mettu and G. Plaxton. The online median problem. *Proceedings of* 41^{st} *IEEE FOCS*, pages 339–348, 2000.

[25] J. Plesnik. A heuristic for the p-center problem in graphs. *Discrete and Applied Mathematics*, 17:263–268, 1987.

[26] D. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems. *Proceedings of* 29^{th} *ACM STOC*, pages 265–274, 1997.

[27] C. Swamy and D. Shmoys. Fault-tolerant facility location. *Proceedings of the* 14^{th} *ACM-SIAM SODA*, pages 735–736, 2003.