Improved Algorithms for Fault Tolerant Facility Location

Sudipto Guha* Adam Meyerson† Kamesh Munagala‡

Abstract

We consider a generalization of the classical facility location problem, where we require the solution to be *fault-tolerant*. Every demand point j is served by r_j facilities instead of just one. The facilities other than the closest one are "backup" facilities for that demand, and will be used only if the closer facility (or the link to it) fails. Hence, for any demand, we 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 this result to 3.16 using randomization and to 2.47 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 [19, 10], plants or warehouses [15, 1, 23], agglomeration of traffic or data [2, 11], among others (refer [8] for a more exhaustive list). The problem, given a set of locations, tries to minimize the sum of the cost of building facilities at a subset of these and the cost of assigning every location 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 linear combination of the costs of assigning a demand location to the two facilities.

In this paper we consider the problem of fault tolerant facility location in which every location j specifies to be assigned to a number r_j of facilities. The cost of assignment of this location

*AT&T Shannon Labs, Florham Park, NJ 07932. Research done while at the Department of Computer Science, Stanford University CA 94305. Research Supported by IBM Research Fellowship, NSF Grant IIS-9811904 and NSF Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation. Email: sudipto@cs.stanford.edu.

†Department of Computer Science, Stanford University CA 94305. 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.

is a weighted combination of these r_j assignments. Recently Jain and Vazirani, [14] provided a primal dual approximation which was logarithmic in the largest requirement[1]. However, the fault tolerant variant of the k–center problem has constant factor approximation algorithms [5, 16, 18, 22]. We resolve this issue by providing a constant factor approximation for the fault tolerant facility location problem. Our result improves [14] even if the maximum requirement is 1. The facility location version is complicated, compared to the fault tolerant k–center problem. Ideas like considering a threshold graph to reduce to the unweighted problem, or searching for a suitable power of a graph to generate a lower bound, do not work. We instead use linear program rounding to provide a constant-approximation - it demonstartes that the natural LP formulation for this problem has smaller integrality gap than any dual formulation.

We use filtering in a fashion similar to [20, 23], however we combine it with scaling and uncrossing steps. These steps allows us to ensure that while we are considering a filtered neighborhood, if a demand point is assigned to several distinct facilities within this neighborhood [2], we will round in such a way that the entire assignment can be rearranged to maintain feasibility.

Towards the later part of the paper 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 [9] to construct a solution of integrality gap 2.47. 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 [17, 3]. The clustered rounding technique of [7, 6] also appears to have this problem with feasibility. We believe that an approximation algorithm for the facility location problem based only on add and exchange operations like in [21] will be the stepping stone for a combinatorial approximation for this problem.

[1] Even for all requirments one, it is at least 3

[2] They cannot be swapped simply: if the first facility has a higher weight in calculating the assignment cost than the backups, for example.

1.1 Previous Results

Classical facility location is MAX-SNP hard [9], and several constant factor approximations [23, 3, 13] are known. Since the problem we study is a generalization of this problem, the hardness results carry over. Many variants of facility location have been studied. The more well known ones include placing capacities on facilities [23, 13], multi-level facility location [1, 10], k-center [12], and k-medians [20, 4, 13, 3]. All these problems have constant factor approximation algorithms.

Jain and Vazirani [14] define the problem of fault tolerant facility location. 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 for this problem. Constant factor approximations are known for the fault-tolerant k-center problem [5, 16, 18, 22], where each demand point j is required to have r_j centers within a fixed distance L from it.

2 Problem Statement

We now present the fault tolerant facility location problem. Just as in classical facility location, we are given a finite metric $G(V,E)$ with a distance function c, a set of possible facility locations $F \subseteq V$, and a set of demand points. Every demand j must be connected to r_j open facilities, but with decreasing weights to further away facilities.

For demand j, let the weights assigned to the links to open facilities in increasing order of distance be $w_j^{(1)} \geq w_j^{(2)} \geq \cdots \geq w_j^{(r_j)}$. 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.

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.

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

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

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. This is not required for the algorithm; this structured solution is constructed so that we can compare our solution to this structured solution, to argue about the approximation factor.

We will treat a demand point j as having r_j copies under the constraint: no two copies of any demand point are assigned to the same facility. In the fractional setting this reduces to the condition $\sum_r x_{ij}^{(r)} \leq y_i \leq 1$.

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 is fixed for the facility for 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, and ensure $\bar{x}_{ij}^{(1)} + \bar{x}_{ij}^{(2)} = y_i$. After that we again pick up one unit of facility fractionally, $\sum_i \bar{x}_{ij}^{(2)} = 1$, and repeat this for all the copies of the demand point. The following lemma is true by construction:

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 *fractionally; thus,* $\int_0^1 \mathcal{C}_j^{(r)}(\beta) d\beta = \mathcal{C}_j^{(r)}$.

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

Lemma 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 with larger weight w (and thus smaller r) go to the closer open facilities.

4 The Algorithm

The algorithm will try to round the fractional solution. It proceeds in two phases. The idea is to use the filtering technique of Lin and Vitter [20] combined with reassignment of the fractional demands, such that each copy of the demand goes to a different facility. Recall, 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 above solution to create a new solution $(\hat{x}, \hat{y})$, which we will round it in the next phase. This phase uses the filtering technique of [20]. For every demand $j^{(r)}$, we consider the fractionally assigned facilities in increasing order of distance. Let i be the first facility in the ordering of $j^{(r)}$ such that $\sum_{i'}^{i} \hat{x}_{i'j}^{(r)} \geq 1-\alpha$. In other words, i is the first facility for which j picks up $1-\alpha$ facility fractionally.

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 $j^{(r)}$. We next set $\hat{y}_i = \min\left\{\frac{y_i}{1-\alpha}, 1\right\}$.

Lemma 4.1 [20] *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 Lemma 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, 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, all the edges used by $j^{(r_1)}$ are at most as long as the edges used by $j^{(r_2)}$ 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

At the beginning of this phase we have a solution $(\hat{x}, \hat{y})$. In this phase we will round the fractional solution. We will perform a rounding similar to [20, 23]. We will preserve $\sum_r \hat{x}_{ij}^{(r)} \leq \hat{y}_i$ as an invariant.

The scheme in [23] does not directly apply, since we have to ensure distinct copies of a demand go 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 perform uncrossing of neighborhoods to ensure 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 the copies (irrespective of demand points) in increasing order of the distance to the farthest fractional facility serving it. We will pick the copies in this order, and repeatedly apply Steps B – E.

- **Step B – Choosing a Facility:** Assume we have picked the r'th copy of the demand point j. Let the set of facilities serving it be $P_j^{(r)}$. Note that copies of j will be picked in increasing order.

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

- **Step C – Merging Facilities:** We now specify which facilities to merge into i:

  1. We select facilities i' with $\hat{x}_{i'j}^{(r)} > 0$ starting with i until the total fraction by which selected facilities are open is exceeds 1. Let $Y = \sum_{i'} \hat{y}_{i'} \geq 1$ be the total fraction by which these facilities are open.
  2. The last picked facility i'' may be picked partially, if $Y > 1$. Since all $\hat{y}_i \leq 1$, $i'' \neq i$. Make two copies, i_1 and i_2, of facility i''. Set $\hat{y}_{i_2} = Y - 1$, and $y_{i_1} = \hat{y}_{i''} - \hat{y}_{i_2}$. For any j' distribute $\hat{x}_{i''j'}^{(r)}$ arbitrarily between the two copies; maintaining $\sum_r \hat{x}_{i'j}^{(r)} \leq \hat{y}_{i'}$ for both $i' = i_1$ and $i' = i_2$. The copy i_1 is selected and i_2 is not. Denote the set of picked facilities by $\hat{P}$.[3]

  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 by $\hat{P}$. We assign the smallest numbered copy (r_1) of j' to be completely served by i. Note that the

[3] $i_1 \in \hat{P}$ and $i_2 \notin \hat{P}$. The logic is that since a facility is being built at i, we can build facility later at i''; but we need to account for $\hat{x}$. Note $\sum_{i' \in \hat{P}} \hat{y}_{i'} = 1$.

assignment distance for $j'^{(r_1)}$ has at most tripled.

- **Step E – Uncrossing Neighborhoods:** We now reassign the remaining copies of j' 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 $\hat{P}$ respectively. Compute $Y_{j'}^{(1)} = \sum_{i' \notin \hat{P}} \hat{x}_{i'j'}^{(r_1)}$, 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. We have the following lemma:

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

  1. $X_{j'}^{(t)} + Y_{j'}^{(t)} = 1$ *for all* $1 \le t \le k$, *and*
  2. $\sum_t X_{j'}^{(t)} \le \sum_{i' \in \hat{P}} \hat{y}_{i'} = 1$.

  Proof: The first part follows from the fact that $\sum_{i'} \hat{x}_{i'j}^{(r)} = 1$ for all $j^{(r)}$. The second part follows from the construction of the set $\hat{P}$. ■

  We now describe the uncrossing step. 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 closest facility 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 $\sum_t \hat{x}_{i'j'}^{(r_t)} \le \hat{y}_{i'}$.

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

  Proof: The only thing we need to check is that we maintain the invariant $\sum_r \hat{x}_{i'j'}^{(r)} \le \hat{y}_{i'}$. Other than this, 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 Lemma 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 [20] and [23], 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 $\hat{\mathcal{C}}_j^{(r)}$, 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.*

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 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:

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

There are only n values of α for which the rounding can be different; our algorithm will choose the best of these n solutions. We present the analysis of the algorithm if α were chosen at random from the interval $(0, 1-x)$. Following analysis of [23], the average cost evaluates to:

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

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 more importantly the similarities between uncapacitated facility location and the fault tolerant version. Consider the heuristic , first analyzed in [9][4]:

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 with best ratio of Gain(i)$/f_i$. If the Gain(i) is positive, the heuristic adds this node 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 prove at every step; the following lemma can be proved:

[4] This actually is known as "Add" heuristic in facility location literature; [9] chose a particular strategy to identify which nodes to add. [3] uses a different strategy (ignoring the delete operations also considered therein).

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

Proof: Consider the fractional solution $(\bar{x}, \bar{y})$, and copy nodes fractionally and sufficiently to ensure that $\bar{x}_{ij}^{(r)} = \bar{y}_i$ or 0 for all i, j, and r. Now consider placing the facilities of the fractional solution (or their copies) with the fraction $\bar{y}_i$; and changing the assignment as mandated by the solution. Define $\text{Gain}'(i)$ to be the change (possibly negative) of cost by this operation. Notice at this step we may be making suboptimal assignments to this fractional facility thus $\text{Gain}'(i) \leq \text{Gain}(i)$. Now

$$\sum_i y_i \text{Gain}'(i) = C - C^* - F^* \quad \text{and} \quad \sum_i y_i f_i = F^*$$

In the first summation, we have to sum over the different copies of a facility since they have separate $\text{Gain}'()$. This shows the average $\text{Gain}'()$ to be as claimed in the lemma, and Gain() being only better, proves the lemma. ■

The above argument is exactly the same for the original facility location problem, as proved in [3]. This is because the fraction solution ensures the fault tolerance by its feasibility; in our solution, we assign the smallest numbered copy (r) of a demand point to this new facility, if closer than the assignment distance for this copy in the current solution. This has a domino effect, since the next smallest copy $(r + 1)$ now gets assigned to where the previous (r'th) copy was assigned to and so forth. Other than the computation of Gain(), the arguments are exact.

It is interesting to note that the combinatorial algorithms proposed in [17, 3] do not extend since deletion of a node cannot be allowed – since it may render a solution infeasible. Both these algorithms employ pure delete operation - where the number of facilities decrease. A combinatorial algorithm that employs add and pure exchange, would very likely extend to the fault tolerant case. In fact the clustered rounding technique in [7, 6] does not extend easily due to the same problem of maintaining feasibility.

The next lemma follows from the above lemma and the analysis in [9], or a simpler one in [3].

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

The above lemma makes the tradeoff in the two costs to be less sharp than what is promised by the rounding scheme. At this point if we assume that we do not have a δ approximation algorithm, over different values of α we can get lower bounds on the assignment cost. Since we know the upper bound of the integral of the assignment cost, for large enough δ, we can show a contradiction. The details of the proof are exactly the same as in [9], and we omit it.

Theorem 4.3 *Fault Tolerant facility location has a 2.47 approximation algorithm in polytime.*

5 Extensions

We can extend the algorithm to incorporate capacities on facilities with buy-at-bulk, as defined in [13]. Facility i has fixed cost f_i and incremental cost per unit demand of δ_i. The incremental cost is computed using the same weights as the routing costs. Note that if we modify the distances so that $c_{ij} = c_{ij} + \delta_i$, the distances still satisfy triangle inequality. Hence a 4 approximation follows.

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*, 1998.

[3] M. Charikar and S. Guha. Improved combinatorial algorithms for facility location and k-median problems. *Proceedings of 40th IEEE FOCS*, 1999.

[4] M. Charikar, S. Guha, D. Shmoys, and É. Tardos. A constant factor approximation algorithm for the k-median problem. *Proceedings of 31st ACM STOC*, 1999.

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

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

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

[8] 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.

[9] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *Proceedings of 9th ACM-SIAM SODA*, 1998.

[10] S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. *Proceedings of 41st IEEE FOCS*, 2000.

[11] S. Guha, A. Meyerson, and K. Munagala. Improved combinatorial algorithms for single sink edge installation problems. *Stanford University Technical Note, STAN-CS-TN-00-96*, 2000.

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

[13] K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. *Proceedings of 40th IEEE FOCS*, 1999.

[14] K. Jain and V. Vazirani. An approximation algorithm for the fault tolerant metric facility location problem. *APPROX*, 2000.

[15] L. Kaufman, M. vanden Eede, and P. Hansen. A plant and warehouse loca-

tion problem. *Operations Research Quarterly*, 28:547–557, 1977.

[16] S. Khuller, R. Pless, and Y. Sussmann. Fault tolerant k-center problems. *Univ. of Maryland Tech. Report, CS-TR-3652*, 1996.

[17] M. R. Korupolu, G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. *Proceedings of 9th ACM-SIAM SODA*, 1998.

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

[19] B. Li, M. Golin, G. Italiano, X. Deng, and K. Sohraby. On the optimal placement of web proxies in the internet. *INFOCOM*, 1999.

[20] J.-H. Lin and J. S. Vitter. ϵ-approximations with minimum packing constraint violations. *Proceedings of 24th ACM STOC*, 1992.

[21] R. Mettu and G. Plaxton. The online median problem. *Proceedings of 41st IEEE FOCS*, 2000.

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

[23] D. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems. *Proceedings of 29th ACM STOC*, 1997.