Improved Combinatorial Algorithms for the Facility Location and k-Median Problems

Moses Charikar* Sudipto Guha†

November 15, 1999

Abstract

We present improved combinatorial approximation algorithms for the uncapacitated facility location and k-median problems. Two central ideas in most of our results are *cost scaling* and *greedy improvement*. We present a simple greedy local search algorithm which achieves an approximation ratio of $2.414+\epsilon$ in $\tilde{O}(n^2/\epsilon)$ time. This also yields a bicriteria approximation tradeoff of $(1+\gamma, 1+2/\gamma)$ for facility cost versus service cost which is better than previously known tradeoffs and close to the best possible. Combining greedy improvement and cost scaling with a recent primal dual algorithm for facility location due to Jain and Vazirani, we get an approximation ratio of 1.853 in $\tilde{O}(n^3)$ time. This is already very close to the approximation guarantee of the best known algorithm which is LP-based. Further, combined with the best known LP-based algorithm for facility location, we get a very slight improvement in the approximation factor for facility location, achieving 1.728. We present improved approximation algorithms for a variant of the capacitated facility location problem. We also present a 4-approximation for the k-median problem, building on and improving the recent 6-approximation due to Jain and Vazirani. The algorithm runs in $\tilde{O}(n^3)$ time.

*`moses@cs.stanford.edu`. Stanford University, Stanford, CA 94305. Research supported by the Pierre and Christine Lamond Fellowship, NSF Grant IIS-9811904 and NSF Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

†`sudipto@cs.stanford.edu`. Stanford University, Stanford, CA 94305. Research Supported by IBM Cooperative Fellowship, NSF Grant IIS-9811904 and NSF Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

1 Introduction

In this paper, we present improved combinatorial algorithms for the facility location and k-median problems. Both problems have been studied extensively and have recently received a lot of attention (see [10, 25, 13, 7, 8, 9, 18] for facility location and [16, 20, 27, 29, 28, 3, 4, 5, 1, 6] for k-median).

Shmoys, Tardos and Aardal [25] gave the first constant factor approximation algorithm for facility location. This was subsequently improved by Guha and Khuller [13], and Chudak and Shmoys [7, 8] who achieved the currently best known approximation ratio of $1 + 2/e \approx 1.736$. All these algorithms are based on solving linear programming relaxations of the facility location problem and rounding the solution obtained. Korupolu, Plaxton and Rajaraman [18] analyzed a well known local search heuristic and showed that it achieves an approximation guarantee of $(5 + \epsilon)$. However, the algorithm has a fairly high running time of $O(n^6 \log n/\epsilon)$.

Until fairly recently, no good approximation algorithms were known for the k-median problem. Lin and Vitter [20] showed that by increasing the number of centers by a factor of $(1 + \epsilon)$, one could obtain a solution whose cost was at most $2(1 + \frac{1}{\epsilon})$ times the cost of the optimal solution (with at most k centers). Their algorithm is based on a technique called *filtering* applied to the solution of a linear programming relaxation to the problem. This filtering technique has since found numerous other applications. Charikar, Guha, Tardos and Shmoys [6] gave the first constant factor approximation algorithm for the k-median problem. This is also based on rounding the solution to a linear programming relaxation.

Very recently, Jain and Vazirani [15] gave primal-dual algorithms for facility location and k-median problems, achieving approximation ratios of 3 and 6 for the two problems. The running time of their algorithms is $O(n^2 \log n)$ for facility location and $O(n^2 \log n(L + \log n))$ for k-median.

1.1 Our Results

We present improved combinatorial algorithms for the facility location and k-median problems. The algorithms combine numerous techniques. However, two ideas are central to most of the algorithms that we obtain. The first is that of *cost scaling* for facility location. The scaling techniques exploits asymmetric approximation guarantees for the facility cost and the service cost. The idea is to apply the algorithm to the *scaled* instance and then scale back to get a solution for the original instance. On several occasions, this technique alone improves the approximation ratios significantly. The second idea used is greedy local improvement. If either the service cost or the facility cost is very high, greedy local improvement decreases the total cost by balancing the two. We show that greedy local improvement by itself yields a very good approximation for facility location in $\tilde{O}(n^2)$ time.

We first present a simple local search algorithm for facility location. This differs from (and is more general than) the heuristic proposed by Kuehn and Hamburger [19], and analyzed by Korupolu *at al.* Despite the seemingly more complex local search step, each step can still be performed in $O(n)$ time. We are not aware of any prior work involving the proposed local search algorithm. We show that the (randomized) local search algorithm together with scaling yields an approximation ratio of $2.414+\epsilon$ in expected time $O(n^2(\log n + \frac{1}{\epsilon}))$ (with an added $\log n$ factor in the running time for a high probability result). This improves significantly on the local search algorithms considered by Korupolu *at al*, both in terms of running time and approximation guarantee. It also improves on the approximation guarantee of Jain and Vazirani, while still running in $\tilde{O}(n^2)$ time. We can also use the local search algorithm with scaling to obtain a bicriteria approximation for the facility cost and the service cost. We get a $(1 + \gamma, 1 + 2/\gamma)$ tradeoff for facility cost versus service cost (within factors of $(1 + \epsilon)$ for arbitrarily small ϵ). Moreover, this holds even when comparing to the cost of an arbitrary feasible solution to the facility location LP. Thus this yields a better tradeoff than the *filtering* technique of Lin and Vitter [20] as well as the tradeoffs in [25, 18]. This gives bicriteria approximations for budgeted versions of facility location where the objective is to minimize either the facility cost or service cost subject to a budget constraint on the other. Further, our tradeoff is close to the best possible. We show that it is not possible to obtain a tradeoff better than $(1+\gamma, 1+1/\gamma)$.

Using both the local search algorithm and the primal dual algorithm of [15], results in a $2\frac{1}{3}$ approximation for facility location in $\tilde{O}(n^2)$ time. We prove that a modified greedy improvement as in [13], along with primal dual algorithm in [15], together with scaling, yields an approximation of 1.853 in $\tilde{O}(n^3)$ time. Interestingly, applying both this combinatorial algorithm (with appropriate scaling) and the LP-based algorithm of [7, 8] and taking the better of the two gives an approximation ratio of 1.728, marginally improving the best known approximation result for facility location. We construct an example to show that the dual constructed by the primal dual facility location algorithm can be a factor $3-\epsilon$ away from the optimal solution. This shows that an analysis that uses only this dual as a lower bound cannot achieve an approximation ratio better than $3-\epsilon$.

Jain and Vazirani [15] show how a version of facility location with capacities (where multiple facilities are allowed at the same location) can be solved by reducing it to an instance of uncapacitated facility location. They obtain a 4 approximation for the capacitated problem in $\tilde{0}(n^2)$ time. Using their idea, from a ρ approximation for the uncapacitated case, one can obtain an approximation ratio of 2ρ for the capacitated case. This was also observed by David Shmoys [23]. This then implies approximation ratios of 3.7 in $\tilde{O}(n^3)$ time and 3.46 using LP-based techniques for the capacitated problem. (The transformation of Jain and Vazirani actually introduces an asymmetric distance function of the form $c'_{ij} = c_{ij} + f(i)$ where the distances c_{ij} form a metric. Most facility location algorithms assume a symmetric distance metric. However, the analysis of the algorithms in question also goes through for asymmetric metrics of the special form constructed.) For lack of space, we will not discuss the results on the capacitated problem.

For the k-median problem, we build on the results of Jain and Vazirani [15] to obtain an approximation ratio of 4, improving on their factor 6 guarantee. Our improvement comes from taking advantage of certain continuity properties of the solution produced by the primal-dual algorithm as the parameter z changes. We also present an increased understanding of the changes in the solution at critical points. Based on this, we modify the algorithm in Jain and Vazirani [15]. The basic idea behind our improvement is again greedy augmentation: We first greedily add centers to the two solutions that we obtain according to a rule that we specify. We present a more sophisticated analysis than that in [15] to show that this yields a 4 approximation for the k-median problem. The running time of our algorithm is $O(n^2 \log n(n+L))$ where L is the number of bits used in the representation of the costs. As in the analysis of Jain and Vazirani, our analysis uses the dual constructed by the algorithm as a lower bound. We construct an example to show that the dual constructed can be a factor $3-\epsilon$ away from the optimal, indicating that an approximation factor better than $3-\epsilon$ cannot be achieved using this approach.

1.2 Problem Definition

The *Uncapacitated Facility Location Problem* is defined as follows: Given a graph with an edge metric $c_{()}$, and cost f_i of opening a center (or facility) at node i, minimize the cost of the selected facilities plus the sum of the assignment cost of each node to its closest open facility. The cost of assigning node j to facility i is $d_j c_{ij}$ where c_{ij} denotes the distance between i and j. The constant d_j is referred to as the demand of the node j.

Due to the metric property, this problem is also referred to as the metric uncapacitated facility location problem. In this paper we will refer to this problem as the facility location problem. We will denote the total cost of the facilities in a solution as the facility cost and the rest as the service cost. They will be denoted by F and C, subscripted appropriately to define the context. For the linear programming relaxations of the problem and its dual see Section 4.

The *k-Median Problem* is defined as follows, given n points in a metric space, we must select k of these to be centers (facilities), and then assign each input point j to the selected center that is closest to it. If location j is assigned to a center i, we incur a cost $d_j c_{ij}$. The goal is to select the k centers so as to minimize the sum of the assignment costs.

We will mostly present proofs showing unit demands; however since the arguments will be on a node per node basis, the results will extend to arbitrary demands as well.

2 Facility Location and Local Search

In this section we describe and analyze a simple greedy local search algorithm for facility location.

Suppose F is the facility cost and C is the service cost of a solution. The objective of the algorithm is to minimize the cost of the solution $F + C$. The algorithm starts from an initial solution and repeatedly attempts to improve its current solution by performing local search operations.

The initial solution is chosen as follows. The facilities are sorted in increasing order of costs. Let F_i be the total facility cost and C_i be the total service cost for the solution consisting of the first i facilities in this order. We compute the F_i and C_i values for all i and choose the solution that minimizes $F_i + C_i$. Lemma 2.1 bounds the cost of the initial solution in terms of the cost of an arbitrary solution SOL and Lemma 2.2 shows that the initial solution can be computed in $O(n^2)$ time.

Let $\mathcal{F}$ be the set of facilities in the current solution. Consider a facility i We will try to improve the current solution by incorporating i and possibly removing some centers from $\mathcal{F}$. (Note that it is possible that $i \in \mathcal{F}$. In fact this is required for reasons that will be made clear later.) Some nodes j may be closer to i than their currently assigned facility in $\mathcal{F}$. All such nodes are reassigned to i. Additionally, some centers in $\mathcal{F}$ are removed from the solution. If we remove a center $i' \in \mathcal{F}$, then all nodes j that were connected to i' are now connected to i. Note that the total change in cost depends on which nodes we choose to connect to i and which facilities we choose to remove from the existing solution. The *gain* associated with i (referred to as gain(i)) is the largest possible decrease in $F + C$ as a result of this operation. If $F + C$ only increases as a result of adding facility i, gain(i) is said to be 0. Lemma 2.3 guarantees that gain(i) can be computed in $O(n)$ time.

The algorithm chooses a random node i and computes gain(i). If gain(i) > 0, i is incorporated in the current solution and nodes are reassigned as well as facilities removed if required so that $C + S$ decreases by gain(i). This step is performed repeatedly. Note that at any point, demand nodes need not be assigned to the closest facility in the current solution. This may happen because the only reassignments we allow are to the newly added facility. When a new facility is added and existing facilities removed, reassigning to the new facility need not be the optimal thing to do. However, we do not re-optimize at this stage as this could take $O(n^2)$ time. Our analysis goes through for our seemingly suboptimal procedure. The re-optimization if required, will be performed later if a facility i already in $\mathcal{F}$ is *added* to the solution by the local search procedure. In fact, this is the reason that node i is chosen from amongst all the nodes, instead of nodes not in $\mathcal{F}$.

We will compare the cost of the solution produced by the algorithm to the cost of an arbitrary feasible solution to the facility location LP (see [25, 7, 15].) The set of all feasible solutions to the LP includes all integral solutions to the facility location problem.

Lemma 2.1 *The cost $F + C$ for the initial solution is at most $n^2 F_{SOL} + nC_{SOL}$ where F_{SOL} and C_{SOL} are the facility costs and service costs in an arbitrary solution SOL to the facility location LP.*

Proof: Consider the solution SOL. Let the $\mathcal{F}$ be the set of facilities i such that $y_i \geq 1/n$. Note that $\mathcal{F}$ must be non-empty. Suppose the most expensive facility in $\mathcal{F}$ has cost f. Then $F_{SOL} \geq f/n$. We claim that every demand node j must draw at least $1/n$ fraction of its service from the facilities in $\mathcal{F}$. Let $C_{\mathcal{F}}(j)$ be the minimum distance of demand node j from a facility in $\mathcal{F}$ and let $C_{SOL}(j)$ be the service cost of j in the solution SOL. Then $C_{SOL}(j) \geq \frac{1}{n} C_{\mathcal{F}}(j)$. Examine the facilities in increasing order of their facility costs and let x be the last location in this order where a facility of cost $\leq f$ occurs. Then the solution that consists of the first x facilities in this order contains all the facilities in $\mathcal{F}$. Thus, the service cost of this solution is at most $\sum_j C_{\mathcal{F}}(j) \leq n \sum_j C_{SOL}(j) = nC_{SOL}$. Also, the facility cost of this solution is at most $x \cdot f \leq n \cdot f \leq n^2 F_{SOL}$. Hence the cost $C + S$ for this solution is at most $n^2 F_{SOL} + nC_{SOL}$. Since this is one of the solutions considered in choosing an initial solution, the lemma follows. □

Lemma 2.2 *The initial solution can be chosen in $O(n^2)$ time.*

Proof: First, we sort the facilities in increasing order of their facility costs. This takes $O(n \log n)$ time. We compute the costs of candidate solutions in an incremental fashion as follows. We maintain the cost of the solution consisting of the first i facilities in this order (together with assignments of nodes to facilities). From this, we compute the cost of the solution consisting of the first $i+1$ facilities (together with assignments of nodes to facilities). The idea is that the solutions for i and $i+1$ differ very slightly and the change can be computed in $O(n)$ time. Consider the effect of including the $(i+1)$st facility in the solution for i. Some nodes may now have to be connected to the new facility instead of their existing assignment. This is the only type of assignment change which will occur. In order to compute the cost of the solution for $i+1$ (and the new assignments), we examine each demand node j and compare its current service cost with the distance of j to the new facility. If it is cheaper to connect j to the new facility, we do so. Clearly, this takes $O(n)$ time.

The initial solution consists of just the cheapest facility. Its cost can be computed in $O(n)$ time. Thus, the cost of the n candidate solutions can be computed in $O(n^2)$ time. The lemma follows. □

Lemma 2.3 *The function $gain(i)$ can be computed in $O(n)$ time.*

Proof: Let $\mathcal{F}$ be the current set of facilities. For a demand node j, let $\sigma(j)$ be the facility in F assigned to j. The maximum decrease in $F + C$ resulting from the inclusion of facility i can be computed as follows. Consider each demand node j. If the distance of j to i is less than the current service cost of j, i.e. $c_{ij} < c_{\sigma(j)j}$, mark j for reassignment to i. Let D be the set of demand nodes j such that $c_{ij} < c_{\sigma(j)j}$. The above step marks all the nodes in D for reassignment to the new facility i. (Note that none of the marked nodes are actually reassigned, i.e. the function σ is not changed in this step) Having considered all the demand nodes, we consider all the facilities in F. Let i' be the currently considered facility. Let $D(i')$ be the set of demand nodes j that are currently assigned to i', i.e. $D(i') = \{j : \sigma(j) = i'\}$. Note that some of the nodes that are currently assigned to i' may have already been marked for reassignment to i. Look at the remaining unmarked nodes (possibly none) assigned to i'. Consider the change in cost if all these nodes are reassigned to i and facility i' removed from the current solution. The change in the solution cost as a result of this is $-f_{i'} + \sum_{j \in D(i') \setminus D}(c_{ij} - c_{i'j})$. If this results in a decrease in the cost, mark all such nodes for reassignment to i and mark facility i' for removal from the solution. After all the facilities in F have been considered thus, we actually perform all the reassignments and facility deletions, i.e. reassign all marked nodes to i and delete all marked facilities. Then gain(i) is simply $(C_1 + S_1) - (C_2 + S_2)$, where C_1, S_1 are the facility and service costs of the initial solution and C_2, S_2 are the facility and service costs of the final solution. If this difference is < 0, gain(i) is 0.

Now we prove that the above procedure is correct. Suppose there is some choice of reassignments of demand nodes and facilities in $\mathcal{F}$ to remove such that the gain is more than than gain(i) computed above. It is easy to show that this cannot be the case. □

Lemmas 2.6 and 2.7 relate the sum of the gains to the difference between the cost of the current solution and that of an arbitrary fractional solution. For ease of understanding, before proving the results in their full generality, we first prove simpler versions of the lemmas where the comparison is to an arbitrary integral solution.

Lemma 2.4 $\sum gain(i) \geq C - (F_{SOL} + C_{SOL})$. *where F_{SOL} and C_{SOL} are the facility and service costs for an arbitrary integral solution.*

Proof: Let $\mathcal{F}_{SOL}$ be the set of facilities in solution SOL. For a demand node j, let $\sigma(j)$ be the facility assigned to j in the current solution and let $\sigma_{SOL}(j)$ be the facility assigned to j in SOL. We now proceed with the proof.

With every facility $i \in \mathcal{F}_{SOL}$, we will associate a modified solution as follows. Let $D_{SOL}(i)$ be the set of all demand nodes j which are assigned to i in SOL. Consider the solution obtained by including facility i in the current solution and reassigning all nodes in $D_{SOL}(i)$ to i. Let $\text{gain}'(i)$ be the decrease in cost of the solution as a result of this modification, i.e. $\text{gain}'(i) = -f_i + \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij})$. Note that $\text{gain}'(i)$ could be < 0. Clearly, $\text{gain}(i) \geq \text{gain}'(i)$. We will prove that $\sum_{i \in \mathcal{F}_{SOL}} \text{gain}'(i) = C - (F_{SOL} + C_{SOL})$. Notice that for $j \in D_{SOL}(i)$, we have $i = \sigma_{SOL}(j)$.

$$\sum_{i \in \mathcal{F}_{SOL}} \text{gain}'(i) = \sum_{i \in \mathcal{F}_{SOL}} -f_i + \sum_{i \in \mathcal{F}_{SOL}} \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{\sigma_{SOL} j})$$

The first term evaluates to $-F_{SOL}$. The summation over the indices $i, j \in D_{SOL}(i)$ can be replaced simply by a sum over the demand points j. The summand therefore simplifies to two terms, $\sum_j c_{\sigma(j)j}$ which evaluates to C, and $-\sum_j c_{\sigma_{SOL} j}$ which evaluates to $-C_{SOL}$. Putting it all together, we get $\sum_i gain'(i) = -F_{SOL} + C - C_{SOL}$, which proves the lemma. $\square$

Lemma 2.5 $\sum gain(i) \geq F - (F_{SOL} + 2C_{SOL})$. *where F_{SOL} and C_{SOL} are the facility and service costs for an arbitrary integral solution.*

Proof: The proof will proceed along similar lines to the proof of Lemma 2.4. Let $\mathcal{F}_{SOL}$ be the set of facilities in solution SOL. For a demand node j, let $\sigma(j)$ be the facility assigned to j in the current solution and let $\sigma_{SOL}(j)$ be the facility assigned to j in SOL. For a facility $i \in \mathcal{F}$, let $D(i)$ be the set of demand nodes assigned to i in the current solution. For a facility $i \in \mathcal{F}_{SOL}$, let $D_{SOL}(i)$ be the set of all demand nodes j which are assigned to i in SOL.

First, we associate every node i' in the current set of facilities $\mathcal{F}$ with its closest node $m(i') \in \mathcal{F}_{SOL}$. For $i \in \mathcal{F}_{SOL}$, let $R(i) = \{i' \in \mathcal{F} | m(i') = i\}$. With every facility $i \in \mathcal{F}_{SOL}$, we will associate a modified solution as follows. Consider the solution obtained by including facility i in the current solution and reassigning all nodes in $D_{SOL}(i)$ to i. Further, for all facilities $i' \in R(i)$, the facility i' is removed from the solution and all nodes in $D(i') \setminus D_{SOL}(i)$ are reassigned to i. Let $\text{gain}'(i)$ be the decrease in cost of the solution as a result of this modification, i.e.

$$\text{gain}'(i) = -f_i + \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) + \sum_{i' \in R(i)} \left(f_{i'} + \sum_{j \in D(i') \setminus D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) \right) \tag{1}$$

Note that $\text{gain}'(i)$ could be < 0. Clearly, $\text{gain}(i) \geq \text{gain}'(i)$. We will prove that $\sum_{i \in \mathcal{F}_{SOL}} \text{gain}'(i) \geq F - (F_{SOL} + 2C_{SOL})$. In order to obtain a bound, we need an upper bound on the distance c_{ij}. From the triangle inequality, $c_{ij} \leq c_{i'j} + c_{i'i}$. Since $i' \in R(i)$, $m(i') = i$, i.e. i is the closest node to i' in $\mathcal{F}_{SOL}$. Hence, $c_{i'i} \leq c_{i'\sigma_{SOL}(j)} \leq c_{i'j} + c_{\sigma_{SOL}(j)j}$, where the last inequality follows from triangle inequality. Substituting this bound for $c_{i'i}$ in the inequality for c_{ij}, we get $c_{ij} \leq 2c_{i'j} + c_{\sigma_{SOL}(j)j} = 2c_{\sigma(j)j} + c_{\sigma_{SOL}(j)j}$, where the equality comes the fact that $i' = \sigma(j)$. Substituting this bound for c_{ij} in the last term of (1), we get

$$\text{gain}'(i) \geq -f_i + \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) + \sum_{i' \in R(i)} \left(f_{i'} + \sum_{j \in D(i') \setminus D_{SOL}(i)} -(c_{\sigma(j)j} + c_{\sigma_{SOL}(j)j}) \right)$$

The last term in the sum is a sum over negative terms, so if we sum over a larger set $j \in D(i')$ instead of $j \in D(i') \setminus D_{SOL}(i)$ we will have a lower bound on $gain'(i)$.

$$\text{gain}'(i) \geq -f_i + \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) + \sum_{i' \in R(i)} f_{i'} - \sum_{i' \in R(i)} \sum_{j \in D(i')} (c_{\sigma(j)j} + c_{\sigma_{SOL}(j)j})$$

The first term in the expression $\sum_i \text{gain}'(i)$ is equal to $-F_{SOL}$, the second is $C - C_{SOL}$ since the double summation over indices i and $j \in D_{SOL}(i)$ is a summation over all the demand nodes j; and $\sum_j c_{\sigma(j)j}$ is C and $\sum_j c_{\sigma_{SOL}(j)j}$ is C_{SOL}. The third term is the sum of the facility costs of all the nodes in the current solution which is F. The fourth term (which is negative) is equal to $-(C + C_{SOL})$ since the summation over the indices $i, i' \in R(i)$ and $j \in D(i')$ amounts to a summation over all the demand nodes j. Therefore we have,

$$\sum_{i \in \mathcal{F}_{SOL}} \text{gain}'(i) \geq -F_{SOL} + (C - C_{SOL}) + F - (C + C_{SOL})$$

which proves the lemma. □

We now generalize Lemma 2.4 to compare with the cost of an arbitrary fractional solution to the facility location LP instead of an integral solution. The facility location LP is as follows:

$$\begin{aligned} \min \sum_i y_i f_i + \sum_{ij} x_{ij} c_{ij} & \\ \forall ij \quad x_{ij} \leq y_i & \\ \forall j \quad \sum_i x_{ij} \geq 1 & \\ x_{ij}, y_i \geq 0 & \end{aligned}$$

Here y_i indicates whether facility i is chosen in the solution and x_{ij} indicates whether node j is serviced by facility i.

Lemma 2.6 *$\sum gain(i) \geq C - (F_{SOL} + C_{SOL})$, where F_{SOL} and C_{SOL} are the facility and service costs for an arbitrary fractional solution SOL to the facility location LP.*

Proof: Consider the fractional solution SOL to the facility location LP. $F_{SOL} = \sum_i y_i f_i$ and $C_{SOL} = \sum_{ij} x_{ij} c_{ij}$. We will prove that $\sum_i y_i \cdot \text{gain}(i) \geq C - (F_{SOL} + C_{SOL})$. Assume without loss of generality that $y_i \leq 1$ for all i and $\sum_i x_{ij} = 1$ for all j.

We first modify the solution (and make multiple copies of facilities) so as to guarantee that for all ij, either $x_{ij} = 0$ or $x_{ij} = y_j$. The number of ij for which this condition is violated are said to be violating ij. The modification is done as follows: Suppose there is violating ij, i.e. there is a facility i and demand node j in our current solution such that $0 < x_{ij} < y_j$. Let $x = \min_j \{x_{ij} | x_{ij} > 0\}$. We create a copy i' of node i (i.e. $c_{i'i} = 0$, $c_{i'j} = c_{ij}$ for all j and $f_{i'} = f_i$.) We set $y_{i'} = x$ and decrease $y_{i'}$ by x. Further, for all j such that $x_{ij} > 0$, we decrease x_{ij} by x and set $x_{i'j} = x$. For all the remaining j, we set $x_{i'j} = 0$. Note that the new solution we obtain is a fractional solution with the same cost as the original solution. Also, all $i'j$ satisfy the desired conditions. Further, the number of violating ij. In at most n^2 such operations, we obtain a solution that satisfies the desired conditions. Suppose $i_1, \ldots, i_r$ are the copies of node i created in this process, The final value of $\sum_{l=1}^{r} y_{i_l}$ is the same as the initial value of y_i.

We now proceed with the proof. For a demand node j, let $\sigma(j)$ be the facility assigned to j in the current solution. With every facility i, we will associate a modified solution as follows. Let $D_{SOL}(i)$ be the set of all demand nodes j such that $x_{ij} > 0$. Note that $x_{ij} = y_i$ for all $i \in D_{SOL}(i)$. Consider the solution obtained by including facility i in the current solution and reassigning all nodes in $D_{SOL}(i)$ to i. Let gain$'(i)$ be the decrease in cost of the solution as a result of this modification, i.e.

$$\text{gain}'(i) = -f_i + \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij})$$

Note that gain$'(i)$ could be < 0. Clearly, gain$(i) \geq$ gain$'(i)$. We will prove that $\sum_i y_i \cdot \text{gain}'(i) = C - (F_{SOL} + C_{SOL})$. Since $y_i = x_{ij}$ if $j \in D_{SOL}(i)$ (also using the fact that $x_{ij} = 0$ if $j \not{i}n D_{SOL}(i)$ to simplify the index of the summation to j), we get,

$$y_i \cdot \text{gain}'(i) = -y_i f_i + \sum_j x_{ij} c_{\sigma(j)j} - \sum_j x_{ij} c_{ij}$$

Summing over all $i \in \mathcal{F}$, the first term evaluates to $-F_{SOL}$, the third term to $-C_{SOL}$. The second term, if we reverse the order of the summation indices, using $\sum_i x_{ij} = 1$ simplifies to $\sum_j c_{\sigma(j)j}$ which evaluates to C, which proves $\sum \text{gain}(i) \geq -F_{SOL} + C - C_{SOL}$. □

Similar to the above lemma, we generalize Lemma 2.5 to apply to any fractional solution of the facility location LP.

Lemma 2.7 $\sum$ *gain*$(i) \geq F - (F_{SOL} + 2C_{SOL})$, *where* F_{SOL} *and* C_{SOL} *are the facility and service costs for an arbitrary fractional solution* SOL *to the facility location LP.*

Proof: Consider the fractional solution SOL to the facility location LP. Let y_i denote the variable that indicates whether facility i is chosen in the solution and x_{ij} be the variable that indicates whether node j is serviced by facility i. $F_{SOL} = \sum_i y_i f_i$ and $C_{SOL} = \sum_{ij} x_{ij} c_{ij}$. We will prove that $\sum_i y_i \cdot \text{gain}(i) \geq F - (F_{SOL} + 2C_{SOL})$. As before, assume without loss of generality that $y_i \leq 1$ for all i and $\sum_i x_{ij} = 1$ for all ij.

Let $\mathcal{F}$ be the set of facilities in the current solution. Mimicking the proof of Lemma 2.5, we will match each node $i' \in \mathcal{F}$ to its "nearest" node in the fractional solution. However, since we have a fractional solution, this matching is a *fractional matching*, given by variables $m_{ii'} \geq 0$ where the value of $m_{ii'}$ indicates the extent to which i' is matched to i. The variables satisfy the constraints $m_{ii'} \leq y_i$, $\sum_i m_{ii'} = 1$ and the values are chosen so as to minimize $\sum_i m_{ii'} c_{ii'}$.

$$\begin{aligned}\sum_i m_{ii'} c_{ii'} &\leq \sum_i x_{ij} c_{ii'} \leq \sum_i x_{ij}(c_{i'j} + c_{ij}) \\ &\leq c_{i'j} + \sum_i x_{ij} c_{ij}.\end{aligned}$$

In particular, for $i' = \sigma(j)$, we get

$$\sum_i m_{i\sigma(j)} c_{i\sigma(j)} \leq c_{\sigma(j)j} + \sum_i x_{ij} c_{ij} \qquad (2)$$

As in the proof of Lemma 2.5, we modify the fractional solution by making multiple copies of facilities such that for every ij, either $x_{ij} = 0$ or $x_{ij} = y_i$ and additionally, for every i, i', either $m_{ii'} = 0$ or $m_{ii'} = y_i$. (The second condition is enforced in exactly the same way as the first, by treating the variables $m_{ii'}$ just as the variables x_{ij}).

For a demand node j, let $\sigma(j)$ be the facility assigned to j in the current solution. Let $D(i')$ be the set of demand nodes j assigned to facility i' in the current solution. With every facility i, we will associate a modified solution as follows. Let $D_{SOL}(i)$ be the set of all demand nodes j such that $x_{ij} > 0$. Let $R(i)$ be the set of facilities $i' \in \mathcal{F}$ such that $m_{ii'} > 0$. Note that $x_{ij} = y_i$ for all $j \in D_{SOL}(i)$ and $m_{ii'} = y_i$ for all $i' \in R(i)$. Consider the solution obtained by including facility i

in the current solution and reassigning all nodes in $D_{SOL}(i)$ to i. Further, for all facilities $i' \in R(i)$, the facility i' is removed from the solution and all nodes in $D(i') \setminus D_{SOL}(i)$ are reassigned to i. Let $\text{gain}'(i)$ be the decrease in cost of the solution as a result of this modification, i.e.

$$\text{gain}'(i) = -f_i + \sum_{j \in D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) + \sum_{i' \in R(i)} \left(f_{i'} + \sum_{j \in D(i') \setminus D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) \right)$$

Clearly, $\text{gain}(i) \geq \text{gain}'(i)$. We will prove that $\sum_i y_i \cdot \text{gain}'(i) \geq F - (F_{SOL} + 2C_{SOL})$. We also know that if $i' \in R(i)$, then $m_{ii'} = y_i$. Since $m_{ii'} = 0$ if i' is not in $R(i)$ we will drop the restiction on i' in the summation.

$$y_i \cdot \text{gain}'(i) = -y_i f_i + \sum_{j \in D_{SOL}(i)} x_{ij}(c_{\sigma(j)j} - c_{ij}) + \sum_{i'} m_{ii'} \left(f_{i'} + \sum_{j \in D(i') \setminus D_{SOL}(i)} (c_{\sigma(j)j} - c_{ij}) \right)$$

From triangle inequality we have $-c_{i'i} \leq c_{i'j} - c_{ij}$. We also replace i' by $\sigma(j)$ wherever convenient. Therefore we conclude,

$$y_i \cdot \text{gain}'(i) \geq -y_i f_i + \sum_{j \in D_{SOL}(i)} x_{ij}(c_{\sigma(j)j} - c_{ij}) + \sum_{i'} \left(m_{ii'} f_{i'} + \sum_{j \in D(i') \setminus D_{SOL}(i)} -m_{i\sigma(j)} c_{i\sigma(j)} \right)$$

Once again evaluating the negative terms in the last summand over a larger set (Namely all $i', j \in D(i')$ instead of $i', j \in D(i') \setminus D_{SOL}(i)$; but since $D(i')$'s are disjoint this simplifies to a sum over j.) and summing the result over all i we have,

$$\sum_i y_i \cdot \text{gain}'(i) \geq -y_i \cdot f_i + \sum_i \sum_{j \in D_{SOL}(i)} x_{ij}(c_{\sigma(j)j} - c_{ij}) + \sum_i \sum_{i'} m_{ii'} f_{i'} - \sum_i \sum_j m_{i\sigma(j)} c_{\sigma(j)i}$$

The first term in the above expression is equal to $-F_{SOL}$, the second term has two parts, the latter of which is $\sum_{i,j} x_{ij} c_{\sigma_{SOL}(j)j}$. Since $i = \sigma_{SOL}(j)$, that is the facility to which j is assigned in the fractional solution, the sum evaluates to the fractional service cost, C_{SOL}. The first part of the second term evaluates to $\sum_j c_{\sigma(j)j}$ since if we reverse the order of the summation, $\sum_{i, j \in D_{SOL}(i)} x_{ij} = 1$, since node j is assigned fractionally. This part evaluates to C.

The third term is equal to $\sum_{i'} f_{i'}$; again reversing the order of the summation and using the fact that $\sum_i m_{ii'} = 1$ from the fractional matching of the nodes i' in the current solution.

We now bound the last term in the expression. Notice the sets $R(i)$ may not be disjoint and we requires a slightly different approach than the proof of lemma 2.5. We use the inequality 2, and the term (which is now $-\sum_i \sum_j m_{i\sigma(j)} c_{\sigma(j)i}$) is at most $-\sum_j c_{\sigma(j)j} - \sum_{i,j} x_{ij} c_{ij}$. The first part evaluates to $-C$ and the second part to $-C_{SOL}$. Substituting these expressions, we get

$$\sum_i y_i \cdot \text{gain}'(i) \geq -F_{SOL} + F + C - C_{SOL} + F - C - C_{SOL}$$

which proves the lemma.

□

Recall that a single improvement step of the local search algorithm consists of choosing a random vertex and attempting to improve the current solution; this takes $O(n)$ time. We now bound the number of improvement steps the algorithm needs to perform till it produces a low cost solution. To begin with, we state and prove a lemma about the stopping time of a certain kind of random process which we will encounter in the analysis.

Lemma 2.8 *Suppose X is a non-negative random variable which takes n possible values which are equally likely and sum up to α. For $t > 0$,*

$$\mathbf{E}[e^{-tX}] \leq e^{\frac{1}{n}(e^{-t\alpha}-1)}.$$

Proof: Suppose the n possible values of X are $x_1, x_2, \ldots, x_n$. Then

$$\mathbf{E}[e^{-tX}] = \frac{1}{n}\sum_{i=1}^{n} e^{-tx_i}.$$

Consider any set of values $\{x_i\}$ such that $x_i \geq 0$, $\sum_{i=1}^{n} x_i = \alpha$ and for which $\sum_{i=1}^{n} e^{-tx_i}$ is maximized. We claim that for any x_i, x_j, $i \neq j$, at least one of x_i, x_j must be 0. If not, we can increase $\sum_{i=1}^{n} e^{-tx_i}$ by replacing the two values x_i, x_j by $0, x_i+x_j$, contradicting the maximality, (Note that $e^{-t\cdot 0}+e^{-t(x_i+x_j)} > e^{-tx_i} + e^{-tx_j}$.) Hence the set of x_i values for which the expression is maximized consists of all x_i's set to 0 except one.

$$\begin{aligned}
\mathbf{E}[e^{-tX}] &= \frac{1}{n}\sum_{i=1}^{n} e^{-tx_i} \\
&\leq \frac{1}{n}(n-1+e^{-t\alpha}) \\
&= 1+\frac{1}{n}(e^{-t\alpha}-1) \\
&\leq e^{\frac{1}{n}(e^{-t\alpha}-1)}.
\end{aligned}$$

□

Lemma 2.9 *Suppose X_i is a sequence of independent random variables such that each X_i has n possible values, each non-negative and equally likely such that the values sum up to at least $\alpha < \beta$. Let S be the smallest value of r for which $\sum_{i=1}^{r} X_i > \beta$. Then*

$$\mathbf{Pr}\left[S > \gamma\frac{n\beta}{\alpha}\right] \leq \left(\frac{\gamma}{e^{(\gamma-1)}}\right)^{\beta/\alpha} \qquad \textit{and} \qquad \mathbf{E}[S] = n\beta/\alpha.$$

Proof: Assume without loss of generality that for each i, the n possible values of X_i sum up to exactly α. We will bound the probability that $S > \gamma n\beta/\alpha$. In order to do this, we derive a Chernoff-like bound. Let $r = \gamma n\beta/\alpha$. Let $X = \sum_{i=1}^{r} X_i$.

$$\begin{aligned}
\mathbf{Pr}\left[S > \gamma\frac{n\beta}{\alpha}\right] = \mathbf{Pr}[S > r] &= \mathbf{Pr}[X < \beta] \\
&= \mathbf{Pr}[e^{-tX} < e^{-t\beta}] \\
&\leq \frac{\mathbf{E}[e^{-tX}]}{e^{-t\beta}} \\
&= \frac{\prod_{i=1}^{r}\mathbf{E}[e^{-tX_i}]}{e^{-t\beta}}
\end{aligned}$$

Here, t is a parameter that will be specified later. By Lemma 2.8, $\mathbf{E}[e^{-tX_i}] \leq e^{\frac{1}{n}(e^{-t\alpha}-1)}$.

$$\mathbf{Pr}\left[S > \gamma\frac{n\beta}{\alpha}\right] \leq e^{\frac{r}{n}(e^{-t\alpha}-1)+t\beta}$$

Setting $t = \frac{1}{\alpha}\ln(\gamma)$, The exponent of the LHS becomes $\frac{\beta}{\alpha}(\ln(\gamma)+1-\gamma)$ which proves the first part.

Now, (using $\beta/\alpha > 1$),

$$\begin{aligned}
\mathbf{E}[S] &\leq \frac{n\beta}{\alpha} + \frac{n\beta}{\alpha}\sum_{\gamma=1}^{\infty}\mathbf{Pr}\left[S > \gamma\frac{n\beta}{\alpha}\right] \\
&\leq \frac{n\beta}{\alpha}\left(1+\sum_{\gamma=1}^{\infty}\gamma\left(\frac{\gamma}{e^{(\gamma-1)}}\right)^{\beta/\alpha}\right) \leq \frac{n\beta}{\alpha}\left(1+\sum_{\gamma=1}^{\infty}\frac{\gamma^2}{e^{\gamma-1}}\right) \\
&= O\left(n\beta/\alpha\right)
\end{aligned}$$

□

We will bound on the number of improvement steps required to reach a low cost solution.

Theorem 2.1 *The algorithm produces a solution such that $F \leq (1+\epsilon)(F_{SOL}+2C_{SOL})$ and $C \leq (1+\epsilon)(F_{SOL}+C_{SOL})$ in expected $O(n(\log n + \frac{1}{\epsilon}))$ steps (running time $O(n^2(\log n + \frac{1}{\epsilon}))$) and in at most $O(n\log n(\log n + \frac{1}{\epsilon}))$ steps (running time $O(n^2\log n(\log n + \frac{1}{\epsilon}))$) with probability $1 - 1/p(n)$ for any polynomial $p(n)$.*

Proof: The proof proceeds roughly by showing that $C + F$ decreases rapidly with every step. For the purpose of analysis, we divide the execution of the algorithm into two phases. The first phase lasts as long as $C + F \geq 2F_{SOL} + 3C_{SOL} + \epsilon(F_{SOL} + C_{SOL})$. After that, we say that we are in the second phase. We terminate the algorithm (for the purpose of analysis) as soon as the conditions in the statement of the lemma are satisfied. We will prove that the expected number of steps in the first phase is $O(n\log(n/\epsilon))$ and the number of steps in the second phase is $O(n/\epsilon)$. At the cost of an extra multiplicative $\log n$ factor, we will also get high probability results.

Consider phase 1. From Lemmas 2.6 and 2.7, we have

$$\sum_i \text{gain}(i) \geq \frac{1}{2}(C + F - (2F_{SOL} + 3C_{SOL}))$$

Let $g(i) = \text{gain}(i)/(C + F - (2F_{SOL} + 3C_{SOL}))$. Then $\sum_i g(i) \geq \frac{1}{2}$. Let P_t be the value of $C + F - (2F_{SOL} + 3C_{SOL})$ after t steps. Let $\text{gain}_t(i)$ and $g_t(i)$ be the values of $\text{gain}(i)$ and $g(i)$ after t steps. Suppose i is the node chosen for step $t+1$. Then,

$$\begin{aligned}
P_{t+1} &= P_t - \text{gain}_t(i) = P_t(1 - g_t(i)) \\
\frac{P_{t+1}}{F_{SOL}+C_{SOL}} &= \frac{P_t}{F_{SOL}+C_{SOL}}(1-g_t(i)) \\
\ln\left(\frac{P_{t+1}}{F_{SOL}+C_{SOL}}\right) &= \ln\left(\frac{P_t}{F_{SOL}+C_{SOL}}\right) + \ln(1-g_t(i)) \\
&\leq \ln\left(\frac{P_t}{F_{SOL}+C_{SOL}}\right) - g_t(i)
\end{aligned}$$

Let $Q_t = \ln(P_t/(F_{SOL}+C_{SOL}))$. Then $Q_{t+1} \leq Q_t - g_t(i)$. Note that the node i is chosen uniformly and at random from amongst n nodes. Let $R_t = g_t(i)$ where i is the node chosen for step $t+1$. Then $Q_{t+1} \leq Q_t - R_t$. R_t is a random variable which can take n values, all equally likely such that the values sum up to at least $\frac{1}{2}$. Note that the initial value $Q_1 \leq \ln n$. Phase 1 ends as soon as $C + F \leq 2F_{SOL} + 3C_{SOL} + \epsilon(F_{SOL} + C_{SOL})$, i.e. as soon as $Q_t < \ln \epsilon$. In other words, phase 1 ends in s steps where s is the smallest integer such that

$$\sum_{t=1}^{s} R_t \geq \ln n - \ln \epsilon = \ln(n/\epsilon).$$

Plugging in $\alpha = 1/2$ and $\beta = \ln(n/\epsilon)$ in Lemma 2.9, the number of steps is $O(n \log(n/\epsilon))$ in expectation and at most $O(n \log n \log(n/\epsilon))$ with probability $1 - 1/p(n)$.

Now consider phase 2. In this phase, $C + F \leq 2F_{SOL} + 3C_{SOL} + \epsilon(F_{SOL} + C_{SOL})$. We stop the analysis as soon as both $C \leq F_{SOL} + C_{SOL} + \epsilon(F_{SOL} + C_{SOL})$ and $F \leq F_{SOL} + 2C_{SOL} + \epsilon(F_{SOL} + C_{SOL})$. Suppose one of these is violated. Then applying either Lemma 2.4 or Lemma 2.5, we get $\sum_i \text{gain}(i) \geq \epsilon(F_{SOL} + C_{SOL})$. Let $S_t = \text{gain}(i)$ where i is the node chosen for step $t+1$ in phase 2. S_t is a random variable with n possible values which are equally likely and sum up to at least $\epsilon(F_{SOL} + C_{SOL})$. The cost $C + F$ goes down by S_t in phase $t+1$. Also the initial value of $C + F$ is at most $2F_{SOL} + 3C_{SOL} + \epsilon(F_{SOL} + C_{SOL})$ and $C + F$ is always non-negative. The number of steps can be bounded by s, where s is the smallest integer such that

$$\sum_{t=1}^{s} S_t \geq 2F_{SOL} + 3C_{SOL} + \epsilon(F_{SOL} + C_{SOL}).$$

Plugging in $\alpha = \epsilon(F_{SOL} + C_{SOL})$ and $\beta = (3 + \epsilon)(F_{SOL} + C_{SOL})$ in Lemma 2.9, the expected number of steps is $O(n/\epsilon)$ and at most $O(n \log n/\epsilon)$ with probability $1 - 1/p(n)$ for any polynomial $p(n)$.

From the analysis of the two phases, number of total steps is $O(n(\log n + \frac{1}{\epsilon}))$ in expectation and at most $O(n \log n((\log n + \frac{1}{\epsilon}))$ with probability $1 - 1/p(n)$ for any polynomial $p(n)$. □

The algorithm can be derandomized very easily, at the cost of a factor n increase in the running time. Instead of picking a random node, we try all the n nodes and choose the node that gives the maximum gain. Each step now takes $O(n^2)$ time. The number of steps required by the deterministic algorithm is the same as the expected number of steps required by the randomized algorithm.

Theorem 2.2 *The deterministic algorithm produces a solution such that $F \leq (1+\epsilon)(F_{SOL} + 2C_{SOL})$ and $C \leq (1+\epsilon)(F_{SOL} + C_{SOL})$ in $O(n(\log n + \frac{1}{\epsilon}))$ steps with running time $O(n^3(\log n + \frac{1}{\epsilon}))$*

3 Scaling Costs

In this section we will show how cost scaling can be used to show a better approximation guarantee. The idea of scaling exploits the asymmetry in the guarantees of the service cost and facility cost.

We will scale the facility costs uniformly by some factor (say δ). We will then solve the modified problem using local search (or in later sections by a suitable algorithm). The solution of the modified instance will then be scaled back to determine the cost in the original instance.

Remark: Notice that the way the proof of theorem 2.1 is presented, it is not obvious what is the termination condition of the algorithm that runs on the scaled instance. We will actually run the algorithm for $O(n(\log n + 1/\epsilon))$ steps and construct that many different solutions. The analysis shows that with constant probability we find a solution such that both the conditions are satisfied for at least one of these $O(n(\log n + 1/\epsilon))$ solutions. We will scale back all these solutions and use the best solution. The same results will carry over with a high probability guarantee at the cost of another $O(\log n)$ in the running time. We first claim the following simple theorem,

Theorem 3.1 *The uncapacitated facility location problem can be approximated to factor $1 + \sqrt{2} + \epsilon$ in randomized $O(n^2/\epsilon + n^2 \log n)$ time.*

Proof: Assume that the facility cost and the service costs of the optimal solution are denoted by F_{OPT} and C_{OPT}. Then after scaling there exists a solution to the modified problem of modified facility cost δF_{OPT} and service cost C_{OPT}. For some small ϵ' we have one solution with service cost C and facility cost F such that

$$F \le (1+\epsilon')(\delta F_{OPT} + 2C_{OPT}) \qquad\qquad C \le (1+\epsilon')(\delta F_{OPT} + C_{OPT})$$

Scaling back, we will have a solution of the same service cost and facility cost F/δ. Thus the total cost of this solution will be

$$C + F/\delta \le (1+\epsilon')\left[(1+\delta)F_{OPT} + (1+\frac{2}{\delta})C_{OPT}\right]$$

Clearly setting $\delta = \sqrt{2}$ gives a $1+\sqrt{2}+\epsilon$ approximation where $\epsilon = (1+\sqrt{2}\epsilon')$. □

Actually the above algorithm only used the fact that there existed a solution of a certain cost. In fact the above proof will go through for any solution, even fractional. Let the facility cost of such a solution be F_{SOL} and its service cost C_{SOL}. The guarantee provided by the local search procedure after scaling back yields facility cost $\hat{F}$ and service cost $\hat{C}$ such that,

$$\hat{F} \le (1+\epsilon)(F_{SOL} + 2C_{SOL}/\delta) \qquad\qquad \hat{C} \le (1+\epsilon)(\delta F_{SOL} + C_{SOL})$$

Setting $\delta = 2C_{SOL}/(\gamma F_{SOL})$ we get that the facility cost is at most $(1+\gamma)F_{SOL}$ and the service cost is $(1+2/\gamma)C_{SOL}$ upto factors of $(1+\epsilon)$ for arbitrary small ϵ. Moreover the guarantee only requires the value of the parameter δ, and not the explicit knowledge of the respective costs. In fact the costs can be guessed upto factors in polynomial in n, and we can try all possible values of the parameter δ upto factors of some $1+\epsilon'$ and we will be guaranteed that some value of δ exists (upto $1+\epsilon'$ factor) such that it will be correct. This factor of $(1+\epsilon')$ can be absorbed by the $1+\epsilon$ term associated with the tradeoff. Thus in some sense we can achieve the tradeoff in an oblivious fashion.

Theorem 3.2 *Let SOL be any solution to the facility location problem (possibly fractional), with facility cost F_{SOL} and service cost C_{SOL}. For any $\gamma > 0$, the local search heuristic proposed (together with scaling) gives a solution with facility cost at most $(1+2/\gamma)F_{SOL}$ and service cost at most $(1+\gamma)C_{SOL}$. The approximation is upto multiplicative factors of $(1+\epsilon)$ for arbitrarily small epsilon.*

The known results on the tradeoff problem use a (p,q) notation where the first parameter p denotes the approximation factor of the facility cost and q the approximation factor for the service cost. This yields a better tradeoff for the k-median problem than the tradeoff $(1+\gamma, 2+2/\gamma)$ given by Lin and Vitter [20] as well as the tradeoff $(1+\gamma, 3+5/\gamma)$ given by Korupolu, Plaxton and Rajaraman [18]. For facility location, our tradeoff is better than the tradeoff of $(1+\gamma, 3+3/\gamma)$ obtained by Shmoys, Tardos and Aardal [25]. We note that the techniques of Marathe *et al* [22] for bicriteria approximations also yield tradeoffs for facility cost versus service cost. Their results are stated in terms of tradeoffs for similar objectives, e.g. (cost,cost) or (diameter,diameter) under two different cost functions. However, their parametric search algorithm will also yield tradeoffs for different objective functions provided there exists a ρ-approximation algorithm to minimize the sum of the two objectives. By scaling the cost functions, their algorithm produces a tradeoff of $(\rho(1+\gamma), \rho(1+1/\gamma))$. For tradeoffs of facility cost versus service cost, ρ is just the approximation ratio for facility location.

The above tradeoff is also interesting since the tradeoff of $(1+\gamma, 1+1/\gamma)$ is the best tradeoff possible. This is illustrated by a very simple example.

3.1 Lower bound for tradeoff

We present an example to prove that the $(1+\gamma, 1+2/\gamma)$ tradeoff between facility and service costs is almost the best possible when comparing with a fractional solution of the facility location LP. Consider the following instance. The instance $\mathcal{I}$ consists of two nodes u and v, $c_{uv} = 1$. The facility costs are given by $f_u = 1$, $f_v = 0$. The demands of the nodes are $d_u = 1$, $d_v = 0$.

Theorem 3.3 *For any $\gamma > 0$, there exists a fractional solution to $\mathcal{I}$ with facility cost F_{SOL} and service cost C_{SOL} such that there is no integral solution with facility cost less than $(1+\gamma)F_{SOL}$ and service cost less than $(1+1/\gamma)C_{SOL}$.*

Proof: Observe that there are essentially two integral solutions to $\mathcal{I}$. The first, SOL_1, chooses u as a facility, $F_{SOL_1} = 1$, $C_{SOL_1} = 0$. The second, SOL_2, chooses v as a facility, $F_{SOL_2} = 0$, $C_{SOL_2} = 1$. For $\gamma > 0$, we will construct a fractional solution for $\mathcal{I}$ such that $F_{SOL} = 1/(1+\gamma)$, $C_{SOL} = \gamma/(1+\gamma)$. The fractional solution is obtained by simply taking the linear combination $(1/(1+\gamma))SOL_1 + (\gamma/(1+\gamma))SOL_2$. It is easy to verify that this satisfies the conditions of the lemma. □

Theorem 3.3 proves that a tradeoff of $(1+\gamma, 1+1/\gamma)$ for facility cost versus service cost is best possible.

4 Primal-Dual Algorithm and Improvements

In this section we will show that the ideas of a primal-dual algorithm can be combined with augmentation and scaling to give a better result than obtained by the pure greedy strategy, and the primal dual algorithm itself.

The facility location problem can be expressed as an IP whose linear relaxation is as follows:

$$\begin{aligned} \min \sum_i y_i f_i + \sum_{ij} x_{ij} c_{ij} & \\ \forall ij \quad x_{ij} &\le y_i \\ \forall j \quad \sum_i x_{ij} &\ge 1 \\ x_{ij}, y_i &\ge 0 \end{aligned}$$

Here y_i indicates whether facility i is chosen in the solution and x_{ij} indicates whether node j is serviced by facility i. Throughout this paper the demand nodes will be indexed by j and facilities by i.

The dual of the facility location LP is:

$$\begin{aligned} \max \sum_j \alpha_j & \\ \forall ij \quad \alpha_j &\le \beta_{ij} + c_{ij} \\ \forall i \quad \sum_j \beta_{ij} &\le f_i \\ \alpha_j, \beta_{ij} &\ge 0 \end{aligned}$$

We describe the primal-dual algorithm (see [15]) very briefly. The graph can be assumed to be a complete bipartite graph between the facilities and the demand nodes. The algorithm maintains dual values for each demand node (α_j) and each edge (β_{ij}). Initially all the variables are set to zero. The algorithm grows the dual variables corresponding to the demand nodes uniformly. At various times some of the dual variables α_j stop growing, to maintain feasibility of the dual solution. We omit the constraints on these dual variables but they are straightforward to infer.

Three types of events occur during the execution of the algorithm. An edge is said to be saturated when the dual variable α_j corresponding to the demand node j is equal to the edge c_{ij} for a facility i. Beyond this point the dual variable β_{ij} corresponding to the edge is raised. The algorithm maintains the invariant that at time t the dual variables α_j that are still being raised are all equal to t and that if $\alpha_j > c_{ij}$ then $\beta_{ij} = \alpha_j - c_{ij}$.

The second type of event is that at some time t_i for a facility i the event that $\sum_j \beta_{ij} = f_i$ where f_i is the facility cost of this facility i. The facility i is declared paid for (This is defined as temporarily open in [15].) and the demand nodes j with positive β_{ij} values stop growing. The node i is defined the *Connecting witness* of j.

The third type of event occurs when for a demand node j the variable $\alpha_j = c_{ij}$ for a paid for (resp. temporarily open) facility i, in which case the variable α_j stops growing. Again node i is the *Connecting witness* of j.

When several events happens at the same time, they are dealt in an arbitrary order. After all the dual variables α_j have stoped growing, the algorithm inspects the subgraph defined by the saturated edges. It recursively picks the facility that was declared paid for the earliest amongst the remaining nodes and builds a center. Then it deletes all paid for nodes that can be reached by two (saturated) edges from this facility.

It may appear strange that we would be using a notion *paid for* instead of the notion already used in [15]. The reason is that the property that a facility is paid for is independent of the declaration whether the facility is *temporarily open* or otherwise. This difference will play an interesting role when we return to primal-dual solutions later.

Lemma 4.1 ([15]) *The primal-dual algorithm returns a solution with facility cost F and a service cost S such that, $3F + S \leq 3OPT$ where OPT denotes the cost of the optimal dual solution.*

In itself the primal-dual algorithm does not yield a better result than factor 3. In fact a simple example shows that the dual constructed which is used as a lower bound to the optimal can be factor 3 away from the true optimum. We will further introduce the notation that the facility cost of the optimal solution be F_{OPT} and the service cost C_{OPT}. *We would like to observe that these quantities are only used in the analysis.*

Let us first consider a simpler algorithm before presenting the best known combinatorial algorithm. If in the primal-dual algorithm we were to scale facility costs by a factor of $\delta = 1/3$ and use primal-dual algorithm on this modified instance we would have a feasible primal solution of cost $F_{OPT}/3 + C_{OPT}$. The primal-dual algorithm giving a solution with modified facility cost F and service cost C will guarantee that $3F + S$ is at most 3 times the feasible dual constructed which is less than the feasible primal solution of $F_{OPT}/3 + C_{OPT}$. After scaling back the solution, the cost of the final solution will be $3F + S$ due to choice of δ, which is at most $F_{OPT} + 3C_{OPT}$.

Now along with this compare the local search algorithm setting $\delta = 2$ for which the cost of the final is $3F_{OPT} + 2C_{OPT}$ from the proof of theorem 3.1. The smaller of the two solutions can be at most $2\frac{1}{3}$ times the optimal which is $F_{OPT} + C_{OPT}$.

Corollary 4.1 *Using augmentation and scaling along with the primal-dual algorithm the facility location problem can be approximated within factor $2\frac{1}{3} + \epsilon$ in time $O(n^2/\epsilon + n^2 \log n)$.*

4.1 Gap examples for dual solutions of primal dual algorithm

We present an example to show that the primal dual algorithm for facility location can construct a dual whose value is $3 - \epsilon$ away from the optimal for arbitrarily small ϵ. This means that it is not possible to prove an approximation ratio better than $3 - \epsilon$ using the dual constructed as a lower bound.

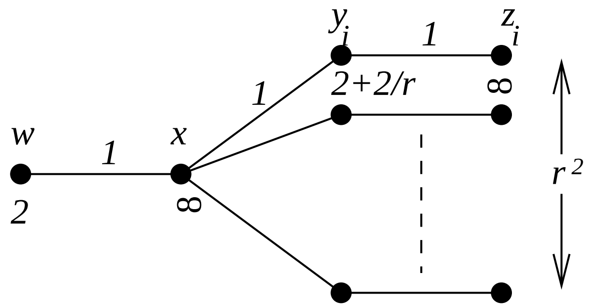

Figure 1: Lower bound example for primal dual facility location algorithm

This lower bound for the primal dual algorithm is the analog of an integrality gap for LPs. We also describe an example that demonstrates a gap of $3-\epsilon$ between the dual constructed by the primal dual k-median algorithm and the optimal solution.

The example is showed in Figure 1. Here r is a parameter. The instance is defined on a tree rooted at w. w has a single child x at unit distance from it. x has r^2 children $y_1, \ldots, y_{r^2}$, each at unit distance from x. Further, each y_i has a single child z_i at unit distance from it. All other distances are shortest path distances along the tree. The node w has a facility cost of 2 and the nodes y_i have facility costs of $2+2/r$; all other nodes have facility costs of ∞. The node x has demand $2r$. Each of the z_i's has unit demand and the rest of the nodes have zero demand. In the dual solution constructed by the algorithm, the α value of x is $2r(1+1/r)$ and the α value for each of the z_i's is $1+2/r$. The value of the dual solution is r^2+4r+2. However, the value of the optimal solution is $3r^2+2r+1$ (corresponding to choosing only w as a facility). The ratio of the optimal solution value to dual solution value exceeds $3-\epsilon$ for r suitably large.

4.2 Greedy strategy

We will use a slightly different augmentation technique as described in [13] and improve the approximation ratio. We will use a lemma proved in [13] about greedy augmentation, however the lemma proved therein was not cast in a form which we can use. The lemma as stated was required to know the value of the optimal facility cost. However the lemma can be proved for any feasible solution only assuming its existence. This also shows that the modification to the LP in [13] is not needed. We provide a slightly simpler proof.

The greedy augmentation process proceeds iteratively. At iteration i we pick a node u of cost f_u such that if the service cost decreases from C_i to C', the ratio $\frac{C_i - C' - f_u}{f_u}$ is maximized. Notice that this is $\text{gain}(u)/f_u$ where $\text{gain}(u)$ is defined as in section 2. That section used the node with the largest gain to get a fast algorithm, here we will use the best ratio gain to get a better approximation factor.

Assume we start with a solution of facility cost F and service cost C. Initially $C_0 = C$. The node which has the maximum ratio can be found in time $O(n^2)$, and since no node will be added twice, the process will take a time at most $O(n^3)$. As an aside we remark that cost of all facilities will be non-zero, otherwise they will always be included in any solution as a preprocessing step.

Lemma 4.2 ([13]) *If SOL is a feasible (fractional) solution, the initial solution has facility cost F and service cost C, then after greedy augmentation the solution is at most*

$$F + F_{SOL} \max\left[0, \ln\left(\frac{C - C_{SOL}}{F_{SOL}}\right)\right] + F_{SOL} + C_{SOL}$$

Proof: If the initial service cost if less than or equal to $F_{SOL}+C_{SOL}$, the lemma is true by observation.

Assume at a point the current service cost C_i is more than $F_{SOL}+C_{SOL}$. The lemma 2.4 guarantees that there is a node with ratio $\frac{C_i-C_{SOL}-F_{SOL}}{F_{SOL}}$. Let the facility cost be denoted by F_i at iteration i. We are guaranteed

$$\frac{C_i - C_{i+1} - (F_{i+1} - F_i)}{F_{i+1} - F_i} \geq \frac{C_i - C_{SOL} - F_{SOL}}{F_{SOL}}$$

This equation rearranges to, (assuming $C_i > C_{SOL} + F_{SOL}$),

$$F_i - F_{i+1} \leq F_{SOL} \left(\frac{C_i - C_{i+1}}{C_i - C_{SOL}} \right)$$

Suppose at the m'th iteration C_m was less than or equal to $F_{SOL} + C_{SOL}$ for the first time. After this point the total cost only decreased. We will upper bound the cost at this step and the result will hold for the final solution. The cost at this point is

$$F_m + C_m \leq F + \sum_{i=1}^{m}(F_i - F_{i-1}) + C_m \leq F + F_{SOL} \left(\sum_{i=1}^{m} \frac{C_{i-1} - C_i}{C_{i-1} - C_{SOL}} \right) + C_m$$

The above expression is maximized when $C_m = F_{SOL} + C_{SOL}$. The derivative with respect to C_m (which is $1 - \frac{F_{SOL}}{C_{m-1}-C_{SOL}}$) is positive since $C_{m-1} > C_{SOL} + F_{SOL}$. Thus since $C_m \leq F_{SOL} + C_{SOL}$ the boundary point of $C_m = F_{SOL} + C_{SOL}$ gives the maxima. In the following discussion we will assume that $C_m = C_{SOL} + F_{SOL}$.

We use the fact that for all $0 < x \leq 1$, we have $\ln(1/x) \geq 1 - x$. The cost is therefore

$$F + \sum_{i=1}^{m} \frac{C_{i-1} - C_i}{C_{i-1} - C_{SOL}} + C_m = F + \sum_{i=1}^{m} \left(1 - \frac{C_i - C_{SOL}}{C_{i-1} - C_{SOL}}\right) + C_m \leq F + \sum_{i=1}^{m} \ln\left(\frac{C_{i-1} - C_{SOL}}{C_i - C_{SOL}}\right) + C_m$$

The above expression for $C_0 = C$ and $C_m = F_{SOL} + C_{SOL}$ proves the lemma. □

4.3 Better approximations for the facility location problem

We now describe the full algorithm. Given a facility location problem we first scale the facility costs by a factor δ such that $\ln(3\delta) = 2/(3\delta)$ as in the section 3. We scale back the solution and apply the greedy augmentation as in [13] and described above. We claim the following.

Theorem 4.2 *The facility location problem can be approximated within factor ≈ 1.8526 in time $O(n^3)$.*

Proof: There exists a solution of cost $\delta F_{OPT} + C_{OPT}$ to the modified problem. Applying the primal-dual algorithm to this gives us a solution of (modified) facility cost F' and service cost C'. If we consider this as a solution to the original problem, then the facility cost is $F = F'/\delta$ and the service cost $C = C'$. Now from the analysis of the primal-dual method we are guaranteed,

$$3\delta F + C = 3F' + C' \leq 3(\delta F_{OPT} + C_{OPT})$$

If at this point we have $C \leq F_{OPT} + C_{OPT}$, then

$$F + C \leq \frac{3\delta F + C}{3\delta} + (1 - \frac{1}{3\delta})C \leq (2 - \frac{1}{3\delta})F_{OPT} + (1 + \frac{2}{3\delta})C_{OPT}$$

Consider the case that $C > F_{OPT} + C_{OPT}$, we also have $C \leq 3\delta F_{OPT} - 3\delta F + 3C_{OPT}$. Since there is a solution of service cost C_{OPT} and facility cost F_{OPT}, by lemma 4.2 the cost is at most,

$$F + F_{OPT}\ \ln\ \left(\frac{3\delta F_{OPT} - 3\delta F + 2C_{OPT}}{F_{OPT}}\right) + F_{OPT} + C_{OPT}$$

Over the allowed interval of F the above expression is maximized at at $F = (2C_{OPT})/(3\delta)$, with cost

$$(1 + \ln(3\delta))F_{OPT} + (1 + \frac{2}{3\delta})C_{OPT}$$

Now for $\delta > 1/3$ the term $1 + \ln(3\delta)$ is larger than $2 - 1/(3\delta)$. Thus the case $C > F_{OPT} + C_{OPT}$ dominates. Consider the value of δ such that $\ln(3\delta) = 2/(3\delta)$, which is $\delta \approx 0.7192$ and the approximation factor is ≈ 1.8526. The greedy augmentation takes maximum $O(n^3)$ time and the primal-dual algorithm takes $O(n^2)$ time. Thus, the theorem follows. □

It is interesting however that this result which is obtained from the above greedy algorithm can be combined with the algorithm of [7, 8] to obtain a very marginal improvement in the approximation ratio for the facility location problem. The results of [7, 8] actually provide a guarantee that if ρ denotes the fraction of the facility cost in the optimal solution returned by the Linear Programming formulation of the facility location problem, then the fractional solution can be rounded within a factor of $1 + \rho \ln \frac{2}{\rho}$ when $\rho \le 2/e$. The above primal-dual plus greedy algorithm when run with $\delta = 2/(3(e-1))$ gives an algorithm with approximation ratio $e - \rho(e - 1 - \log \frac{2}{e-1})$. It is easy to verify that the smaller of the two solutions has an approximation of 1.728. This is a very marginal (≈ 0.008) improvement over the best known algorithm. However it demonstrates that the facility location problem can be approximated better.

Theorem 4.3 *Combining the linear programming approach and the scaled, primal-dual plus greedy algorithms, the facility location problem can be approximated to a a factor* 1.728.

5 Augmentation in k-Median problem

We present a modification of the algorithm in [15] and improve the approximation factor to 4. In Section 5.1, we prove several continuity properties of the dual solution produced by the primal-dual facility location algorithm. We also define a slightly different notion of ordering of events. In Section 5.2, we describe the modification to the algorithm in [15]. In Section 5.3, we analyze the modified algorithm. In Section 5.4 we determine the running time of the algorithm.

5.1 Properties of the Dual Solution

In this subsection we will prove certain continuity properties of the solution produced by the primal-dual facility location algorithm (referred to simply as 'the algorithm' for brevity). These continuity properties will be crucial to the analysis of the modified algorithm later. The dual variables α_j and β_{ij} will represent the dual variables corresponding to vertices and edges as in the previous sections. Let $\alpha_j(z)$ be the final value of α_j produced by the algorithm for z. Similarly, define $\beta_{ij}(z)$. We define $t_i(z)$ be the last time at which $\sum_j \beta_{ij}$ increases in the algorithm for z. Note that $\sum_j \beta_{ij}$ is the total contribution to the facility building cost of i. When the algorithm is run for a particular value of z, $\sum_j \beta_{ij} \le z$. If the final value of $\sum_j \beta_{ij}$ is indeed z, then we say that facility i is *paid for* and t_i is the time at which facility i is paid for.

The ordering at z refers to the pair $(\mathcal{F}, \mathcal{S})$ where $\mathcal{F}$ is the set of facilities that are paid for in the algorithm for z and $\mathcal{S}$ is the ordered (non-decreasing) sequence of $\alpha_j(z)$ values, $t_i(z)$ values and c_{ij} values, such that consecutive elements of this ordering are related either by equality or strict inequality. We define a *transition point* to be a value of z where infinitesimal changes in z result in a change in the ordering. This definition of transition point is similar to the definition of a critical point in [15]; note however that our definition is independent of the ordering used by the algorithm to resolve ties. A transition point can be defined in terms of $t_i(z)$ values as well, our definition in terms of $\alpha_j(z)$ values is purely for convenience. Analogous to Lemma 10 in [15], we can prove the following lemma about the minimum separation between two transition points:

Theorem 5.1 *Two critical values of z are separated by at least $c = 2^{-(poly(n)+L)}$, where L is the number of bits needed to represent the longest edge.*

Proof: The idea is to show that a transition point is the infimum or supremum of a set which is the feasible region of a polynomial sized linear program. □

We will show that all the functions $\alpha_j(z)$ are continuous. We will also be interested in the times at which nodes are declared open. In fact we will we will extend the above notion of time when a facility is declared open to all nodes and a notion when a node is last contributed to towards building a center. Denoting these quantities as a function of z, say $t_u(z)$ (The formal definition is provided later), we will be interested in the continuity of these functions. Though we will not be able to show that these functions are continuous, we will be able to show piece-wise continuity and hence the existence of left and right limit points of these functions at a critical value.

First let us discuss the continuity of the functions $\alpha_j(z)$.

Continuity of $\alpha_j(z)$ We will prove that $\alpha_j(z)$ is continuous by considering two specific values of z namely z' and $z' + \delta z$ and δz is infinitesimally small. We will show that the values of $\alpha_j(z')$ and $\alpha_j(z' + \delta z)$ are at most a finite multiple of $|\delta z|$ apart. This will imply that $\alpha_j(z)$ is continuous.

Denote by $\alpha_j(z, t)$, the value of dual variable α_j at time t in the algorithm for z. It is easy to observe that $\alpha_j(z, t) = \min(t, \alpha_j(z))$. Define $p_u(z, t) = \sum_j \max(0, \alpha_j(z, t) - c_{uj})$ as the total contribution to the facility building cost of u at time t in the algorithm corresponding to z. A node is declared temporarily open at time t, if and only if $p_u(z, t) = z$.

Theorem 5.2 *The dual variables $\alpha_j(z)$ is continuous in z, moreover for a small change δz in value of z, the value of $\alpha_j(z)$ changes by at most $2^n|\delta z|$.*

Proof: Suppose we execute separately and in parallel, the algorithm for z' and $z'' = z' + \delta z$. An *event* is said to occur if a node j gets a connecting witness in one of the two executions. Thus we have $2n$ events in all, two corresponding to each node j. The events for node j occur at times $\alpha_j(z')$ and $\alpha_j(z'')$.

Consider the set of values of $\alpha_j(z')$ and $\alpha_j(z'')$. Sort them in non-decreasing order and break ties arbitrarily. In all there are $2n$ events in this order. Let the i'th event correspond to node j. Look at the set of nodes corresponding to the first $i - 1$ events.

We will prove that for the i'th event, the $\alpha_j(z)$ values of the concerned node differ by at most $f(i)\delta z$. Where the function $f(i) = \sum_r^{i-1} f(r)$. We will proceed by induction. The hypothesis is clearly true for $i = 1$, because no node has been considered yet. We will show that the hypothesis holds for the i'th event, for the node j in question as well. Without loss of generality assume that it was the $\alpha_j(z')$ value. If the value $\alpha_j(z'')$ has been considered already then there is nothing to prove. Therefore we assume that the value $\alpha_j(z'')$ occurs after this event. The node j's dual value got fixed at this value because it got a connecting witness u at the time $t = \alpha_j(z')$. Therefore in the solution corresponding to z', either at this point u was declared open, or $t = \alpha_j(z') = c_{uj}$ and u was declared open already. In both cases $t \geq c_{ij}$. Also since the $\alpha_j(z)$ values are increased uniformly for other nodes, not involved in the first $i-1$ events, the current dual value in the order corresponding to both the orders, at time t is exactly t.

We have $p_u(z', t) = \sum_j \max(0, \alpha_j(z', t) - c_{uj}) = z'$ at this point. For all the nodes involved in the first $i - 1$ events, their $\alpha_{j'}(z)$ values are perturbed. For all other nodes the current value of the dual solution in both the orders is the same (t). Therefore, $p_u(z'')$ is at least $z' - \sum_r^{i-1} f(r)\delta z$.

If the node u is not declared temporarily open, then clearly in time at most $t + \sum_r^{i-1} f(r)\delta z + \delta z$ the value of $p_u(z'')$ will be z'' unless $\alpha_j(z'')$ stops growing. So in the ordering corresponding to z'', certainly j will get a connecting witness at or before this time. Some other node with a tight edge to j may be declared temporarily open, but in all cases $\alpha_j(z'')$ is at most $\alpha_j(z) + \sum_r^{i-1} f(r)\delta z + \delta z$. Setting $f(i) = \sum_r^{i-1} f(r) + 1$, we prove the induction hypothesis for all cases.

Now since $f(0) = 0$, we can upper bound $f(i)$ by 2^i. In any case since $f(i)$ is finite, in the limit $\delta z \to 0$, we have $\alpha_j(z') = \alpha_j(z'')$. Hence we can conclude that $\alpha_j(z)$ is continuous. □

5.2 Modified Algorithm

As in the algorithm of [15], we will use a primal dual algorithm for facility location to obtain a solution to the k-median problem. We set the cost of each facility to be z and run the primal dual algorithm for the resulting facility location problem for different values of z.

The order in which events occur, together with the order in which ties are resolved determines the primal solution produced at the end of of the algorithm. The dual solution produced is independent of the order in which ties are resolved. A binary search on the value of z yields two values z_1, z_2 which are very close such that for $z = z_1$, the algorithm produces $\hat{k}_1 < k$ centers (the *small* solution) and for $z = z_2$, the algorithm produces $\hat{k}_2 > k$ centers (the *large* solution). We will carry on the binary search till the values of z are sufficiently "close". We will discuss this in section 5.4 later.

The Augment Step: We incorporate an *augmentation* step which adds facilities if possible in the small and large solutions. Note that this step is performed only after the small and large solutions have been found for two very close values of z; *this is not performed during the binary search.*

We describe how the small solution (corresponding to z_1) is augmented. The augmentation of the large solution (corresponding to z_2) is done in a symmetric fashion other than the stopping condition.

1. Consider all nodes which are centers in the large solution and are not centers in the small solution. Let i be the currently considered node.
2. If the small solution has k centers, stop.
3. If there exists a node i' which is chosen as a center in the current small solution (which resulted from a previous augmentation) and demand node j such that $\beta_{ij}(z_1) > 0, \beta_{ij}(z_2) > 0, \beta_{i'j}(z_1) > 0, \beta_{i',j}(z_2) > 0$, i cannot be added. Otherwise add node i to the small solution.
4. Repeat the above steps until the solution has k nodes or all nodes are considered.

Denote the augmented small solution by $\mathsf{soln}^{(1)}$ and the augmented large solution by $\mathsf{soln}^{(2)}$. Let them open k_1 and k_2 centers respectively. By construction, $k_1 \le k$ and $k_2 \ge k$. The final solution to the k-median problem will be either $\mathsf{soln}^{(1)}$ or $\mathsf{soln}^{(2)}$ or a subset of centers in either of the solutions, under a suitable distribution. Before we prove the approximation factor for the above algorithm we discuss some of the properties of the dual solution.

5.3 Analysis

The analysis of the performance of the above algorithm will be divided into two parts. In the first part we will demonstrate certain key properties in the two solutions found after the augment step is complete. We will use the lemmata proved in the first part to prove the approximation factor. The next lemma is from [15]. However it was not cast in such a form in [15].

Lemma 5.1 *For any solution constructed by the primal-dual algorithm (say at value z) and any demand node j there exists node u at distances at most $3\alpha_j(z)$ from j. Moreover $t_u(z) \le \alpha_j(z)$.*

Proof: Suppose u' is the connecting witness of node j in the orders corresponding to z, then $t_{u'}(z) \leq \alpha_j(z)$ and $c_{u'j} \leq \alpha_j(z)$.

If u' was chosen as a center in the solution (while choosing an independent set, before the augmentation step) then $u = u'$ and the lemma is true.

If u' was not chosen as a center, then there exists a center u in the solution such that $t_u(z) \leq t_{u'}(z)$, and there exists a node j' such that $c_{uj'}$ and $c_{u'j'}$ are both less than $\alpha_{j'}(z)$. Also for this node j' we have

$$\alpha_{j'} \leq t_u(z) \leq t_{u'}(z) \leq \alpha_j(z)$$

Now by triangle inequality $c_{uu'} \leq 2\alpha_j(z)$ and by another application $c_{uj} \leq c_{uu'} + c_{u'j} \leq 3\alpha_j(z)$. Therefore the lemma follows. □

The next lemma is a consequence of the augment step, and the central piece of the improvement.

Lemma 5.2 *If $k_1 < k$, for every node u in $\mathsf{soln}^{(2)}$ (resp. $\mathsf{soln}^{(1)}$) there exists a center in $\mathsf{soln}^{(1)}$ (resp $\mathsf{soln}^{(2)}$) which is at most $2\min(t_u(z_1), t_u(z_2))$ away from u.*

Proof: If u were included in $\mathsf{soln}^{(1)}$, then the lemma is true.

Since u was not included in $\mathsf{soln}^{(1)}$ and that $k_1 < k$, u was checked for augmentation in $\mathsf{soln}^{(1)}$. Thus there exists a vertex i and a vertex v such that,

- v is a center in $\mathsf{soln}^{(1)}$,
- $\beta_{vi}(z_1),\ \beta_{vi}(z_2),\ \beta_{ui}(z_1),\ \beta_{ui}(z_2)$ are positive.

Since $\beta_{vi}(z_1)$ and $\beta_{vi}(z_2)$ are positive, we have $c_{vi} < \alpha_i(z_1)$ and $c_{vi} < \alpha_i(z_2)$. Similarly we have c_{ui} less than both $\alpha_i(z_1)$ and $\alpha_i(z_2)$. Again since $\beta_{ui}(z_1)$ is positive, we have $\alpha_i(z_1) \leq t_u(z_1)$ and similarly $\alpha_i(z_2) \leq t_u(z_2)$. Thus we can bound c_{vi} and c_{ui} by $\min(\alpha_i(z_1), \alpha_i(z_2))$.

By triangle inequality,

$$c_{uv} \leq c_{iv} + c_{iu} < 2\min(\alpha_i(z_1), \alpha_i(z_2)) \leq 2\min(t_u(z_1), t_u(z_2))$$

For the case of $u \in \mathsf{soln}^{(1)}$ the argument is exactly the same, except the condition $k_1 < k$ is not required since all the nodes in $\mathsf{soln}^{(1)}$ are candidates for augmentation in $\mathsf{soln}^{(2)}$. □

Lemma 5.3 *For any node u in $\mathsf{soln}^{(1)}$ (or $\mathsf{soln}^{(2)}$), either $\sum_j \beta_{uj}(z_1) = z_1$ or $\sum_j \beta_{uj}(z_2) = z_2$. If there is a single transition point z^* between z_1 and z_2, we have $\sum_j \beta_{uj}(z^*) = z^*$.*

Proof: The second part of the claim follows from the fact that if u is temporarily open in z_1 (resp. z_2) then it is temporarily open for all values of z between z^* and z_1 (resp. z_2). □

Lemma 5.4 *For every demand node j there exists at most one node u in $\mathsf{soln}^{(1)}$ (resp. $\mathsf{soln}^{(2)}$) such that $\min(\beta_{uj}(z_1), \beta_{uj}(z_2)) > 0$. In particular, if there exists a single transition point z^* between z_1 and z_2, then for every demand node j there exists at most one node u in $\mathsf{soln}^{(1)}$ (resp. $\mathsf{soln}^{(2)}$) such that $\beta_{uj}(z^*) > 0$.*

Proof: We prove the lemma for $\mathsf{soln}^{(1)}$, for $\mathsf{soln}^{(2)}$ the arguments are exactly the same.

Suppose not, then there exists two such nodes u and v. We have $\beta_{uj}(z_1)$, $\beta_{uj}(z_2)$, $\beta_{vj}(z_1)$, $\beta_{vj}(z_2)$ all positive.

Consider the node (amongst u and v) which it was included later. It cannot be included in phase 2 corresponding to z_1, since the edges (u,j) and (v,j) both were tight at z_1 (since the $\beta_{uj}(z_1)$ and $\beta_{vj}(z_1)$ are positive). It also cannot be included by the augment step since all $\beta_{uj}(z_1)$, $\beta_{uj}(z_2)$, $\beta_{vj}(z_1)$, and $\beta_{vj}(z_2)$ are positive. Therefore for all cases, u and v could not together be in $\mathsf{soln}^{(1)}$.

Assuming there is exactly one transition point between z_1 and z_2 (including them), if $\beta_{uj}(z^*)$ is positive, then by continuity, in a small open interval around z^*, we have $\beta_{uj}(z) > 0$. Now the fact that $\beta_{uj}(z_1) > 0$ means that $\alpha_j(z_1)$ is strictly greater than c_{uj}. Therefore in the semi-open interval $(z^*, z_1]$ if for one point z we have $\beta_{uj}(z) > 0$, then $\beta_{uj}(z)$ positive throughout. Which is a consequence of the ordering of $\alpha_j(z)$ and the c_{uj} remaining invariant. Likewise for $z \in [z_2, z^*)$. Therefore, $\beta_{uj}(z_1)$ and $\beta_{uj}(z_2)$ are both positive, and the lemma follows by the above argument.

Likewise we can argue that for $\mathsf{soln}^{(2)}$. □

5.3.1 Approximation Factor

We will give a solution to the k-median problem based on $\mathsf{soln}^{(1)}$ and $\mathsf{soln}^{(2)}$. In fact, we will give a probability distribution on solutions, but this can be derandomized easily by the method of conditional probabilities. In this subsection we will assume that there is *exactly one* transition point between z_1 and z_2. Denote the transition point by z^*. We will argue for that value of $z = z^*$ in the dual solution. In the later part of this section, we will show how to remove that restriction.

Let A_j be the assignment cost of j, i.e. the distance to the nearest facility in our solution. A_j will be a random variable as mentioned before. We will assign a facility opening cost q_j to j, where $\sum_j q_j = zk$. The proof of the following theorem is straightforward.

Theorem 5.3 *If for some z, $\sum_j q_j = zk$ and for all ji, $\mathbf{E}[A_j] + c \cdot q_j \leq c \cdot \alpha_j(z)$ then the expected cost of the solution is a factor c times the optimal.*

We need to distinguish the cases when the augment step considered all nodes or not. This corresponds to $k_1 < k$ and $k_1 = k$. We consider a definition first.

Definition 1 *If for a node j there exists a node u in $\mathsf{soln}^{(1)}$ such that $\min(\beta_{uj}(z_1), \beta_{uj}(z_2))$ is positive, define the node to be* directly assigned *to u in $\mathsf{soln}^{(1)}$. Otherwise define j to be* indirectly assigned *in $\mathsf{soln}^{(1)}$. Likewise define directly or indirectly assigned in $\mathsf{soln}^{(2)}$.*

Case $k_1 = k$ or $k_2 = k$. We can claim that we actually have a factor 3 approximation for the k-median problem. Assume $K_1 = k$. If a node j is directly assigned to u in $\mathsf{soln}^{(1)}$, then we set $q_j = \beta_{uj}(z^*)$ otherwise $q_j = 0$. The claim follows for $k_2 = k$ analogously. See 5.5.

Case $k_1 < k$ and $k_2 > k$ Consider the following final solution, with $a = \frac{k_2 - k}{k_2 - k_1}$ and $b = \frac{k - k_1}{k_2 - k_1}$.

1. For every node in $\mathsf{soln}^{(1)}$, find the nearest center in the $\mathsf{soln}^{(2)}$.
2. Choose the pair with the smallest distance.
3. Remove both nodes (for now) and repeat the above until every node in $\mathsf{soln}^{(1)}$ is paired up.
4. For each of the k_1 pairs, independently build a center at the node corresponding to $\mathsf{soln}^{(1)}$ with probability a otherwise build a center at the node corresponding to $\mathsf{soln}^{(2)}$ (with probability b).
5. Of the remaining $k_2 - k_1$ nodes in $\mathsf{soln}^{(2)}$ which are not paired up, choose a subset of $k - k_1$ nodes independently at random and build a center there.
6. Assign every node to its nearest center.

Lemma 5.5 *If $k = k_1$ then $\mathsf{soln}^{(1)}$ gives a 3 approximation for the k-median problem.*

Proof: By lemma 5.4 we have for any node j, $\beta_{uj}(z^*)$ is positive for at most one node u. Therefore,

$$\beta_{uj}(z^*) = \sum_{u \in \mathsf{soln}^{(1)}} \beta_{uj}(z^*)$$

Summing over all j, and using the fact that by lemma 5.3 we have $\sum_j \beta_{uj}(z^*) = z^*$, we have

$$\sum_j q_j = \sum_j \beta_{uj}(z^*) = \sum_j \left(\sum_{u \in \mathsf{soln}^{(1)}} \beta_{uj}(z^*) \right) = \sum_{u \in \mathsf{soln}^{(1)}} \left(\sum_j \beta_{uj}(z^*) \right) = \sum_{u \in \mathsf{soln}^{(1)}} z^* = k_1 z^*$$

Suppose j was directly assigned to u in $\mathsf{soln}^{(1)}$. In this case $A_j = c_{uj}$. Therefore $A_j + q_j = c_{uj} + \beta_{uj}(z^*) = \alpha_j(z^*)$, moreover since $q_j \leq \alpha_j(z^*)$, we have

$$A_j + 3q_j \leq 3\alpha_j(z^*)$$

If j was indirectly assigned in $\mathsf{soln}^{(1)}$. By lemma 5.1 we have a center u in $\mathsf{soln}^{(1)}$ before augmentation (hence after it as well) such that $c_{uj} \leq 3\alpha_j(z_1)$. Since $q_j = 0$ in this case,

$$A_j + 3q_j \leq 3\alpha_j(z_1)$$

Using continuity arguments as $z \to z^*$, for both cases we have

$$A_j + 3q_j \leq 3\alpha_j(z^*)$$

Now by theorem 5.3 we have a 3 approximation for the k-median problem. □

Each center in $\mathsf{soln}^{(2)}$ is present with probability at least b. Also every node in $\mathsf{soln}^{(1)}$ is present with probability a. In the rest of this subsection we will be use case by case analysis for every proof. We will consider the type of a node j defined as follows. II: Indirectly assigned in solutions $\mathsf{soln}^{(1)}$ and $\mathsf{soln}^{(2)}$; DI: Directly assigned to u in $\mathsf{soln}^{(1)}$ and indirectly in $\mathsf{soln}^{(2)}$; and likewise ID and DD. We will set $q_j = q_j(1) + q_j(2)$. If j is assigned directly to u in $\mathsf{soln}^{(1)}$, set $q_j(1) = a\beta_{uj}(z^*)$ else $q_j(1) = 0$. Similarly $q_j(2) = b\beta_{uj}(z^*)$ if j is directly assigned to u in $\mathsf{soln}^{(2)}$ and 0 otherwise.

Lemma 5.6 *For the solution described above, for every node j the following is satisfied;*

$$E[A_j] + c \cdot q_j \leq c\alpha_j(z^*) \qquad\qquad \sum_j q_j = kz^*$$

where $c = \max(2(1+a), 2(1+b), 3 + 2ab + 2a^2b^2)$.

Proof: The charge q_j has essentially two parts corresponding to direct assignment or otherwise in the solutions. Denote them by $q_j(1)$ and $q_j(2)$ Using a very similar proof as in the first part of lemma 5.5, we can show that over all the direct assignment in $\mathsf{soln}^{(1)}$, we have $\sum_j q_j(1) = ak_1 z^*$, and similarly $\sum_j q_j(2) = bk_2 z^*$. The proofs are exactly the same as in the lemma 5.5 with factors a and b present appropriately. Since $ak_1 + bk_2 = k$, it follows that $\sum_j q_j = kz^*$.

We will prove the rest of the lemma case by case on the type of a demand node j.

Case DD: Assume the node j was of type DD. Let j be assigned directly to u and v in solutions $\mathsf{soln}^{(1)}$ and $\mathsf{soln}^{(2)}$ respectively. If u and v are paired up then the expected service cost of j is at most $ac_{uj} + bc_{vj}$. If they are not paired, then assuming u got paired up first, if node u is not present then it definitely has a center within distance $c_{uj} + c_{vj}$ from it. For this case node u is present with probability a conditioned on the fact that u is not present, v is present with probability $(1-a)b$ (Since it is independent of the conditioning.) Also there exists a center within distance $2c_{uj} + c_{vj}$ from j if none are present.

So we first try to assign the node to u, if u is not present then we try to assign the node to v, and if neither is then the center to which u is paired to. The expected service cost for this case is

$$E[A_j] \leq ac_{uj} + b^2 c_{vj} + ab(2c_{uj} + c_{vj})$$

Analogously had v been paired up first, the expected cost is at most

$$E[A_j] \leq bc_{vj} + a^2 c_{uj} + ab(2c_{vj} + c_{uj})$$

The RHS of both the equations can be bound by

$$(a + 2ab)c_{uj} + (b + 2ab)c_{vj}$$

In this case $cq_j = ca\beta_{uj}(z^*) + cb\beta_{vj}(z^*)$. Therefore for this case we have

$$E[A_j] + cq_j \leq (a+2ab)c_{uj} + (b+2ab)c_{vj} + ca\beta_{uj}(z^*) + cb\beta_{vj}(z^*) \leq ca(c_{uj} + \beta_{uj}(z^*)) + cb(c_{vj} + \beta_{vj}(z^*))$$

The second part of the inequality follows since both $1 + 2b$ and $1 + 2a$ are at most 3. The RHS sums to $c(a + b)\alpha_j(z^*)$, which proves the lemma for this case.

Case DI. Consider the node u to which j is assigned to directly in $\mathsf{soln}^{(1)}$. We will have two cases to consider depending on t_u and α_j.

Sub-case $\alpha_j(z_1) \geq t_u(z_1)$: By lemma 5.2 there exists a node v in $\mathsf{soln}^{(2)}$ such that,

$$c_{uv} \leq 2\min(t_u(z_1), t_u(z_2)) \leq 2\alpha_j(z_1)$$

So if the node u is paired with v or u is paired off before v then there exists a center within distance $2\alpha_j(z_1)$ of u if u is not present. The expected service cost for these two cases is

$$E[A_j] \leq ac_{uj} + b(c_{uj} + 2\alpha_j(z_1))$$

If v was paired off first, then u is present with probability a, and conditioned on u not being present, v is present with probability $(1-b)b$ at a distance $c_{uj} + 2\alpha_j(z_1)$. If both of them are not present, (wit h probability ab) then there is a center at a distance $2\alpha_j(z_1)$ from v, which is at most $4\alpha_j + c_{uj}$ from j. We first try to assign j to u, if u is not present then to v and if neither then to the center v is paired to. Thus the expected distance can be bounded by

$$ac_{uj} + b^2(2\alpha_j(z_1) + c_{uj}) + ab(4\alpha_j(z_1) + c_{uj}) = c_{uj} + 2b(1 + a)\alpha_j(z_1)$$

In this case $q_j = a\beta_{uj}(z^*)$, therefore we have,

$$E[A_j] + cq_j \leq (\max(1, ca))\alpha_j(z^*) + 2b(1 + a)\alpha_j(z_1)$$

We argue that the term $\alpha_j(z_1)$ can be made $\alpha_j(z^*)$, which is a consequence of taking limits $z \to z^*$ over the semi-open interval $(z_1, z^*]$ and the continuity of $\alpha_j(z)$.

It is easy to verify that $\max(1, ca) + 2b(1 + a)$ is at most c for $a, b > 0$ with $a + b = 1$.

Sub-case $\alpha_j(z_1) < t_u(z_1)$: By applying lemma 5.1 for both the solutions constructed at $z = z_1$ and $z = z_2$, we have that there exists nodes u_1 in $\mathsf{soln}^{(1)}$ and u_2 in $\mathsf{soln}^{(2)}$ at distances at most $3\alpha_j(z_1)$ and

$3\alpha_j(z_2)$ from j respectively. Moreover $t_{u_1}(z_1) \leq \alpha_j(z_1)$ and $t_{u_2}(z_2) \leq \alpha_j(z_2)$. From this it is clear that $u \neq u_1$.

If $u_1 = u_2$ or that u_1 is paired to u_2 it is easy to show the required bound of $E[A_j]$. Similarly if $u_1 \in \mathsf{soln}^{(2)}$ and $u_2 \in \mathsf{soln}^{(1)}$. Assume that $u_1 \neq u_2$ and $u_1 \notin \mathsf{soln}^{(2)}$ and $u_2 \notin \mathsf{soln}^{(1)}$.

Applying lemma 5.2 on u_1, there exists u_3 in $\mathsf{soln}^{(2)}$is at most a distance $2t_{u_1}(z_1)$ by lemma which is at most $2\alpha_j(z_1)$. If $u_2 = u_3$ or $u_3 \in \mathsf{soln}^{(1)}$ then for node j, it first tries node u, then any of u_1 or u_2, and if none of them (u_1 or u_2) is present, then the pair of u_1 or u_2 whichever got paired first.

$$E[A_j] \leq ac_{uj} + b[\max(3\alpha_j(z_1), 3\alpha_j(z_2)) + 2ab\alpha_j(z_1)]$$

For the case $u_2 \neq u_3$, assuming that u_1 was paired off before u_3 (u_3 need not be paired off) the pair of u_1 is at most $2\alpha_j(z_1)$ from u_1 and the above equation can be used to bound $E[A_j]$. If u_3 were paired off first, the pair of u_3 can be at most $2\alpha_j(z_1)$ from u_3 since at the time u_3 was paired off, u_1 was a possible candidate for the pair.

Now applying lemma 5.2 on u_2 there exists u_4 in $\mathsf{soln}^{(1)}$which is distance $2\alpha_j(z_2)$ from u_2. If $u_4 = u$ or u_1 or $u_4 \in \mathsf{soln}^{(2)}$ then again the above equation can be derived (changing some $\alpha_j(z_1)$ terms to $\alpha_j(z_2)$ which we will see is not crucial).

Thus the case we are left to consider is $\{u, u_1, u_2, u_3, u_4\}$ are all distinct and u_3 paired off before u_1. In this case we assign j to u, if u is not present to u_1 or u_2. If both u_1 and u_2 are not present, we try u_3 or u_4. If both u_3 and u_4 are absent, we try the pair of u_3.

$$E[A_j] \leq ac_{uj} + b[\max(3\alpha_j(z_1), 3\alpha_j(z_2)) + ab[(1-ab)\max(2\alpha_j(z_1), 2\alpha_j(z_2)) + ab4\alpha_j(z_1)]]$$

Since $\alpha_j(z_1)$ and $\alpha_j(z_2)$ are $\alpha_j(z^*)$ at the critical point, for this sub-case we have

$$E[A_j] + cq_j \leq ca(c_{uj} + \beta_{uj}(z^*)) + b(3 + 2ab + 2a^2b^2)\alpha_j(z^*) \leq ca\alpha_j(z^*) + cb\alpha_j(z^*)$$

Remark: There is a small dependency in the probability of u_2 and u_4, however since we are interested in a lower bound of the probability of one being present when the other is not, the dependency actually benefits the inequality .

Case II: Assuming j was of type II we make the same argument as in the latter sub-case of type DI. Except that in this case the node u is not present and $q_j = 0$.

$$E[A_j] + cq_j \leq (3 + 2ab + 2a^2b^2)\alpha_j(z^*) + 0$$

Case ID: Assuming j was of type ID, we can proceed exactly along the lines of the case DI interchanging the role of $\mathsf{soln}^{(1)}$ and $\mathsf{soln}^{(2)}$. We will again have two sub-cases, for the first one we will have

$$E[A_j] + cq_j \leq (\max(1, cb))(c_{uj} + \beta_{uj}(z^*)) + 2a(1+b)\alpha_j(z^*)$$

And for the latter sub case the expression derived will be

$$E[A_j] + cq_j \leq cb\alpha_j(z^*) + a(3 + 2ab + 2b^2a^2)\alpha_j(z^*)$$

Thus the lemma is true for all cases. □

5.4 Running Time

The previous subsection essentially showed that the modified algorithm provides a factor 4 approximation if the binary search is carried on till one single transition point is left. In this subsection we will argue in a similar fashion as in [15] that the binary search can be terminated when the difference $\Delta(z) = |z_1 - z_2|$ is very small. Define $\Delta(\alpha) = 2^n \Delta(z)$.

However for the purpose of the proof, we used the fact that the $\beta_{ij}(z)$ values are close (for $z = z_1, z_2, z^*$). In fact we would require that the difference in their values be small, that is $1/poly(n)$ of the optimal solution, for a suitably large $poly(n)$. The caveat is that to ensure that the $\beta_{ij}(z)$ values are polynomially close, the z values have to be exponentially close. In fact from the proof of continuity of the $\alpha_j(z)$ values we know that $\Delta(z)$ must have precision $1/(2^{n+L} poly(n))$ for some polynomial $poly(n)$. This essentially says that the binary search needs to be carried on at least $O((n+L)\log n)$ steps. Thus we have a $O(n^2(n+L)\log n)$ algorithm ($n^2 = m$).

Theorem 5.4 *If for some z and $\Delta(\alpha) = c_{min}/(16n^2)$ (c_{min} being the smallest non-zero edge)*

$$\sum_j q_j = zk \quad \text{and} \quad \mathbf{E}[A_j] + c \cdot q_j \leq c \cdot \alpha_j(z) + 4\Delta(\alpha)$$

then the expected cost of the solution is a factor $c + 2/n$ times the optimal.

Proof: Summing the first equation over all nodes j, we get,

$$\begin{aligned} \sum_j E[A_j] + ckz &\leq \sum_j (E[A_j] + c \cdot q_j) \leq c \sum_j \alpha_j(z) + 4n\Delta(\alpha) \\ \sum_j E[A_j] &\leq c\left(\sum_j \alpha_j(z) - zk\right) + 4n\Delta(\alpha) \end{aligned}$$

If the optimal solution is non-zero, it is clearly at most c_{min}. The case where the optimal solution is cost 0 is trivial. Therefore we can say that $E[A_j] \geq c_{min} \geq 16n^2\Delta(\alpha)$. (This implies $\Delta(z) = c_{min} 2^{-n-4}/n^2$.) Thus we have

$$\left(1 - \frac{1}{4n}\right) E[A_j] \leq c\,(\alpha_j(z) - zk)$$

Since $\sum_j \alpha_j(z) - zk$ is the value of the dual solution to the k-median problem, this implies that the expected cost of the solution is at most a factor $c + 2/n$ times the optimal, using $c < 4$. □

The idea of the rest of the proof is simple. We will show the above conditions of the theorem are true at $z = z^*$. Notice that we do not need to know the values q_j. The q_j values will be defined as in the previous section. (Actually the definition will depend on $k_1 = k$ or otherwise.) Thus part of the conditions of the theorem are already met.

Lemma 5.7

$$\mathbf{E}[A_j] + c \cdot q_j \leq c \cdot \alpha_j(z^*) + 4\Delta(\alpha)$$

Proof: Consider the proofs of lemma 5.5 and lemma 5.6. By inspection we can conclude that

$$\mathbf{E}[A_j] + c \cdot q_j \leq c \cdot \max\{\alpha_j(z_1), \alpha_j(z_2), \alpha_j(z^*)\}$$

Now all these $\alpha_j(z)$ values differ from $\alpha_j(z^*)$ by $2^n\Delta(z)$, which is $\Delta(\alpha)$ and c is at most 4. □

The only other observation we require is that c is at most $4 - 2/k$, since either a or b has to be at least $1/k$.

Therefore we can prove the following theorem,

Theorem 5.5 *There is a 4 approximation for the k-median problem in time $O(n^2(L+n)\log n)$*

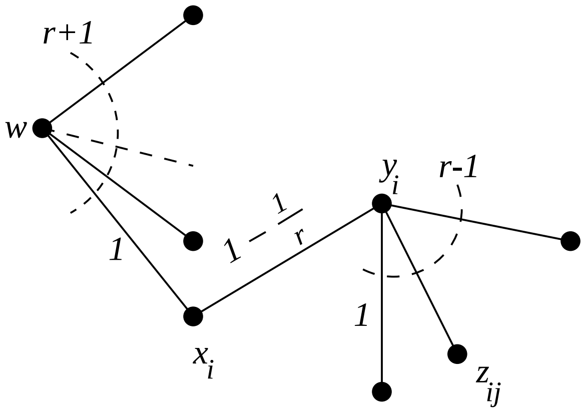

Figure 2: Lower bound example for primal dual k-median algorithm

5.5 Lower bound for the primal dual method for k-median

The lower bound example for k-median is shown in Figure 2. Again r is a parameter. The instance is defined on a tree rooted at w. w has $r+1$ children $x_1, \ldots, x_{r+1}$. Each node x_i has a single child y_i. Further, each node y_i has $r-1$ children $z_{i1}, \ldots, z_{i(r-1)}$. The distance between w and x_i is 1, that between x_i and y_i is $1 - 1/r$, and the distance between y_i and z_{ij} is 1. All other distances are the shortest path distances along this tree. The nodes x_i and z_{ij} have unit demands; all other nodes have zero demand. For this instance $k = 2$. The transition point occurs at $z = \theta = 1 + 1/r$. For $z = \theta^-$, all the y_i nodes are chosen as facilities; this solution has $r+1$ centers. For $z = \theta^+$, only w is chosen as a center; this solution has 1 center. At the transition point all the α values are $1 + 1/r$. The value of the dual solution is $r^2 + O(r)$. However, the value of the optimal solution is $3r^2 - O(r)$. Thus the ratio between the two exceeds $3 - \epsilon$ for arbitrarily small ϵ by choosing r sufficiently large.

Acknowledgements

We would like to thank Chandra Chekuri, Ashish Goel, Samir Khuller, Rajeev Motwani, David Shmoys and Eva Tardos for numerous discussions on these problems. We would also like to thank Kamal Jain and Vijay Vazirani for informing us of their results.

References

[1] S. Arora, P. Raghavan and S. Rao. Approximation schemes for Euclidean k-medians and related problems. *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp. 106–113, 1998.

[2] J. Bar-Ilan, G. Kortsarz, and D. Peleg. How to allocate network centers. *J. Algorithms*, 15:385–415, 1993.

[3] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, pages 184–193, 1996.

[4] Y. Bartal. On approximating arbitrary metrics by tree metrics. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 161–168, 1998.

[5] M. Charikar, C. Chekuri, A. Goel, and S. Guha. Rounding via trees: deterministic approximation algorithms for group Steiner trees and k-median. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 114–123, 1998.

[6] M. Charikar, S. Guha, É. Tardos and D. B. Shmoys. A constant factor approximation algorithm for the k-median problem. To appear in *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, 1999.

[7] F. A. Chudak. Improved approximation algorithms for uncapacitated facility location. In R. E. Bixby, E. A. Boyd, and R. Z. Ríos-Mercado, editors, *Integer Programming and Combinatorial Optimization*, volume 1412 of *Lecture Notes in Computer Science*, pages 180–194, Berlin, 1998. Springer.

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

[9] F. A. Chudak and D. B. Shmoys. Improved approximation algorithms for the capacitated facility location problem. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages S875–S876, 1999.

[10] G. Cornuéjols, G. L. Nemhauser, and L. A. Wolsey. 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.

[11] M. E. Dyer and A. M. Frieze. A simple heuristic for the p-center problem. *Oper. Res. Lett.*, 3:285–288, 1985.

[12] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoret. Comput. Sci.*, 38:293–306, 1985.

[13] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 649–657, 1998.

[14] D. S. Hochbaum and D. B. Shmoys. A best possible approximation algorithm for the k-center problem. *Math. Oper. Res.*, 10:180–184, 1985.

[15] K. Jain and V. Vazirani, Primal-Dual Approximation Algorithms for Metric Facility Location and k-Median Problems. Manuscript, March 1999. Available at `http://www.cc.gatech.edu/fac/Vijay.Vazirani/k-median.ps`

[16] O. Kariv and S. L. Hakimi. An algorithmic approach to network location problems, Part II: p-medians. *SIAM J. Appl. Math.*, 37:539–560, 1979.

[17] S. Khuller and Y. J. Sussmann. The capacitated k-center problem. In *Proceedings of the 4th Annual European Symposium on Algorithms, Lecture Notes in Computer Science 1136*, pages 152–166, Berlin, 1996. Springer.

[18] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–10, 1998.

[19] A. A. Kuehn and M. J. Hamburger. A heuristc program for locating warehouses. *Management Science*, 9:643–666, 1979.

[20] J.-H. Lin and J. S. Vitter. Approximation algorithms for geometric median problems. *Inform. Proc. Lett.*, 44:245–249, 1992.

[21] J.-H. Lin and J. S. Vitter. ϵ-approximations with minimum packing constraint violation. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 771–782, 1992.

[22] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. .Rosenkrantz and H. B. Hunt III. Bicriteria Network Design Problems. *Journal of Algorithms*, 28(1):142–171, 1998.

[23] D. B. Shmoys. *personal communication*, April 1999.

[24] D. B. Shmoys and É. Tardos. An Approximation Algorithm for the Generalized Assignment Problem. *Math. Programming*, 62:461–474, 1993.

[25] D. B. Shmoys, É. Tardos, and K. I. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 265–274, 1997.

[26] M. Sviridenko. Personal communication, July, 1998.

[27] A. Tamir. An $O(pn^2)$ algorithm for the p-median and related problems on tree graphs. *Oper. Res. Lett.*, 19:59–94, 1996.

[28] S. de Vries, M. Posner, and R. Vohra. The K-median Problem on a Tree. *Working paper*, Ohio State University, Oct. 1998.

[29] J. Ward, R. T. Wong, P. Lemke, and A. Oudjit. Properties of the tree K-median linear programming relaxation. Unpublished manuscript, 1994.