Facility Location in a Global View

Wenqiang Dai, Peng Xiao, and Ke Jiang

School of Management, Xi'an Jiaotong University,
Xi'an, 710049, P.R. China
wqdai@mail.xjtu.edu.cn

Abstract. Facility Location Problems have always been studied with the assumption that the environment in the network is static and does not change over time. In practice, however, the environment is usually dynamic and we must consider the facility location in a global view. In this paper, we impose the following additional constraints on input facilities: the total number of facilities to be placed is not known in advance and a facility cannot be removed once it is placed. We solve this problem by presenting an algorithm to find a facility permutation such that any prefix of the permutation of facilities is near-optimal over any other facility subset.

1 Introduction

Variants of the facility location problem (FLP) have been studied extensively in operation research and management science literatures [14, 3, 1]. The model in typical theoretical work has addressed situations in which we want to locate facility in a network and optimize an objective function in the static environment. In practice, however, the environment is dynamic in many cases. For example, in a commercial network, we do not know the exact number of facility in advance, our business plan is to start with one facility, and then to gradually add a new facility but never to remove a previously established facility. The same is true for locating the public facility in community network, such as schools, hospitals, etc.

Under the above considerations, the facilities must be constructed in the global view. That is, we locate facilities in a way that their cost is optimal or near-optimal over any other non-empty facility subset and, when the number of needed facilities increases, no former facility can be removed. It is worth noting that a typical facility location problem with these two constraints can be viewed as the facility permutation problem whose goal is to specify the facility order that minimizes the maximum ratio between the cost of any prefix of the permutation and that of an any non-empty subset of all facilities.

Of the facility location models, our problem most closely resembles Undesirable Facility Location Problem [13], the Online Median Problem [11] and the Incremental Facility Location Problem [12]. All of these problems, as well as our problem, used exactly the same assumption that the number of facilities is not known in advance

N. Megiddo, Y. Xu, and B. Zhu (Eds.): AAIM 2005, LNCS 3521, pp. 142–150, 2005.

and once the facility is placed, it can't be removed, but their objective formulations are different with ours. See the section 2 for some more discussions.

In this paper, we solve the facility location problem with above considerations by defining a model of locating facilities to optimize the location cost over any other cost of non-empty subset of all facilities, which we call strength facility location problem (SFLP). Although NP-hardness of SFLP has not been proven yet, finding an optimal solution for the problem seems to be difficult, in view of the NP-completeness of the relaxation version: classical Uncapacitated Facility Location Problem. This paper is concerned with approximation algorithm for this problem. Given a minimization problem, an algorithm is said to be a (polynomial) $r - approximation$ algorithm, if for any instance of the problem, the algorithm runs in polynomial time and produces a solution that has a cost at most $r \geq 1$ times the minimal cost, where r is called the *approximation ratio* of the algorithm.

Though not stated specifically, Mettu and Plaxton [11] first presented a 3-approximation algorithm for SFLP. But their solution does not generate the permutation of facilities. Based on their algorithm, this paper presents a group of simple and deterministic approximation algorithms for constructing some solutions of SFLP which is similar to, in simplicity and efficiency, the standard heuristics for facility location. Each approximation algorithm and the result are only linked to a single parameter, and for the different parameter, the solutions obtained by each algorithm are *nested*, namely, one of solution is the subset of another one, thus we obtain a permutation of all facilities by varying the parameter.

The rest of this paper is organized as follows. Section 2 specifies the problem and describes related works. Section 3 presents the approximation algorithm and proves the approximation ratio. The final section, section 4, concludes the paper and describes future research.

2 Problem Description and Related Work

This section describes the formulation of SFLP and gives an overview of related work. We first discuss the basic facility location model, the metric uncapacitated facility location problem (UFLP), in which we are given a graph with nonnegative edge costs. As motivation [7], the nodes can be thought of as customers, the facilities as service centers, and the distance between a customer and a service center as the cost of serving the customer by that center. Furthermore, each customer (node) has a weight, corresponding to the amount of requests. So, the cost of serving a customer becomes the weight of the customer node times its distances from the closest service center. In addition, each node is assigned a constructive cost to represent the cost of building a service center at that node. The total cost we wish to minimize is total constructive cost of chosen facilities plus the cost of serving all of the customer requests by the chosen facilities. The objective of the UFLP is to choose k nodes (as the facilities) so as to minimize the cost, where k is a given facility number. Compared with this, the goal of

SFLP is to minimize the cost over the cost of any number of facilities. More precisely, the problem formulation of SFLP is as follows:

Problem Formulation: Fixed a set of points U, a distance function $d : U \times U \longrightarrow \mathrm{R}^+$, a nonnegative weight function $w : U \longrightarrow \mathrm{R}$ and nonnegative construtive function $f : U \longrightarrow \mathrm{R}$. We assume that the distance that d is a metric, that is, d is nonnegative, symmetric, satisfies the triangle inequality, and $d(x, y) = 0$ iff $x = y$. We define the distance of a point x to a point set S is $d(x, S) = \min_{y \in S} d(x, y)$ and $|S|$ is the number of points in S. Let $n = |U|$ denote the number of total customers, and for any subset $S \subseteq U$, let its cost be $cost(S) = \sum_{x \in S} f(x) + \sum_{y \in U} d(y, S)w(y)$. In contrast to UFLP whose objective is to find a subset $S \subseteq U$ such that it meet $\min_{|S|=k} cost(S)$, where k is the given positive integer, $1 \leq k \leq n$, the objective of SFLP is to give a subset $S \subseteq U$ such that it meet that $\min_{1 \leq k \leq n} \min_{|S|=k} cost(S)$.

Related Work: Facility location has been the subject of a great deal of previous work[5, 15, 9, 8, 2, 6], and here we just describe some typical theoretical analysis. This problem is MAX-$\mathcal{SNP}$ Hard and the first constant approximation algorithm was given by Shmoys et.al [15]; the approximation ratio was later improved to 1.728 by Charikar and Gula[5] and to 1.528 by Sviridenko [16]. Now the best approximation ratio is 1.52 given by Mahdian, Ye and Zhang [10], and the negative result is that no polynomial-time algorithm can achieve an approximation ratio less than 1.463 unless $\mathbf{NP} \subseteq \mathrm{DTIME}[n^{O(\log \log n)}]$ [8]. The methods they have used are based on, such as linear Programming rounding (e.g. [15]), local search (e.g. [5]), and the primal-dual method (e.g. [9]) etc. The reader is referred to Mahdian et.al [10] for a detail discussion.

Current et.al [4] firstly considered the facility location problem when the number of facilities is uncertain. Unlike ours, they used the criteria of the minimization of expected opportunity loss and the minimization of maximum regret to make decision about what's the number of facilities and where we locate those. Their solution, unfortunately, does not also generate a permutation of facilities and thus does not meet the demand that no point can be removed when the number of facilities increases.

Recently, the Undesirable Facility Location Problem [13], the Online Median Problem [11] and the Incremental Facility Location Problem [12] most resemble our problem, but their objective formulations are different with ours. The Undesirable Facility Location Problem seeks a solution to maximize the minimum distance between facilities and the minimum distance between facilities and existing non-obnoxious facilities [13]. In the Online Median Problem, the goal is to determinate a permutation of facilities that minimizes the maximum ratio between the service cost of any prefix of the permutation and that of an optimal *offline same-size* configurations [11], and this problem uses the formulation $cost(S) = \sum_{y \in U} d(y, S)w(y)$ to compute the cost. And for the Incremental Facility Location Problem [12], the cost formulation is $cost(S) = |S| \sum_{x \in S} f(x) + \sum_{y \in U} d(y, S)w(y)$ and the goal is also minimal over the *same-size* configurations.

3 Algorithm and Approximation Ratio

3.1 Greedy Selection of Mettu and Plaxton [11]

For the new requirement of our problem, we first present the algorithm of Mettu and Plaxton [11], and will need to build upon it. Roundly speaking, their algorithm is to compute the "value" of each ball about every node in the metric space to start with, then sort them by increasing order, then greedily pick up the point if they separate sufficient large, and so on until all n points are examined. These points are taken as service facilities. If the distance function is a metric and the separation distance is more twice longer than the maximal radius of ball, the cost of resulting location is within a factor three of optimal. Implicitly, their idea came from the work of Jain and Vazirani [9].

The following definition were used in [11], but we rewrite them by our notation for easy use later.

Definition 1. *A* **ball** *A is a pair (x, r_x), where $x \in U$ is the center and r_x is the radius of the ball, which is a nonnegative real.*

Definition 2. *Given a ball $A = (x, r_x)$, we let* **Points(A_x)** *denote the set $\{y \in U | d(x, y) \leq r_x\}$ and always directly use A_x instead of Point(A_x). For example, we write "$a \in A_x$" and "$A_x \cup A_y$" instead of "$a \in Points(A_x)$" and "$Points(A_x) \cup Points(A_y)$", respectively.*

Definition 3. *The* **value** *of a ball $A = (x, r_x)$, denoted $value(A_x)$, is defined by*

$$value(A_x) = \sum\nolimits_{y \in A_x} (r_x - d(x, y)) \cdot w(y)$$

Definition 4. *For any ball $A = (x, r_x)$ and any nonnegative real c, we define* **cA** *as the ball (x, cr_x).*

3.2 Constructing a Permutation

We will stick with the same computing of "value" of each point in the metric space, but we will no longer greedily select the facility by the former separate distance. Note that the longer the separate distance between two facilities is, the less the number of obtained facilities is. Intuitively, we may only need the distance to separate longer than twice and thus it will enable us to generate less facilities. Thus if the obtained facilities are nested each other and the near-optimal property is still held for fixed constant distance, the permutation of facilities, which will meet the constraints of our problem, will be obtained. Following we will give a positive answer for these considerations.

For easy reference, and to illustrate our notation, we write integrally the algorithm. In the following algorithm and the later analysis, we always assume that the weight of each point is larger than zero for the sake of convenience.

Input: (U, d), f and w;

Output: A non-empty subset $Z_n \subseteq U$.

Algorithm:

Step 1. For each point x, determine an associated ball $A = (x, r_x)$ such that $value(A_x) = f(x)$.

Step 2. Sorting r_x for all $x \in U$ increasingly, denoting the index of x after sorting is $\phi(x)$.

Step 3. Let $B_i = (x_i, r_{x_i})$ denote the ball $A = (x, r_x)$ such that $\phi(x) = i$, $0 \leq i < n$. Let $Z_0 = \emptyset$.

Step 4. For $i = 0$ to $n - 1$: If $Z_i \cap (1 + \alpha)B_i = \emptyset$ then let $Z_{i+1} = Z_i \cup \{x_i\}$; otherwise, let $Z_{i+1} = Z_i$.

Where $\alpha \geq 1$ is a parameter which need to be inputed.

Throughout the rest of the paper, we always let Z denote the result of our algorithm.

Remark 1. We have $x_0 \in Z$, so $Z \neq \emptyset$.

Remark 2. Note that if $\phi(x) = i$, we have $value(B_i) = value(A_{x_i}) = f(x_i)$.

Remark 3. Following the same manner as analysis given in [11], it can be easily obtained that the total time complexity of above algorithm is $O(n^2)$ time for any fix constant $\alpha \geq 1$.

3.3 Performance Guarantee

We now show a strong guarantee for the facility location induced by our algorithm with any fixed input $\alpha \geq 1$.

Lemma 1. *For any point $x \in U$, there exists a point $y \in Z$ such that $\phi(y) \leq \phi(x)$ and $d(x, y) \leq (1 + \alpha)r_x$.*

Proof. If $x \in Z$, we can choose $y = x$ and this lemma is proven. Following we assume $x \notin Z$.

The proof is by contradiction. Assume that $\forall y \in Z$ with $\phi(y) \leq \phi(x)$, we have $d(x, y) > (1 + \alpha)r_x$, that is, $Z_{\phi(x)} \cap (1 + \alpha)B_{\phi(x)} = \emptyset$. Thus we have x belongs to Z according to step 4 and this is contradicted with $x \notin Z$. □

Lemma 2. *Let $x, y \in Z$ and $x \neq y$, then $d(x, y) > (1 + \alpha)max\{r_x, r_y\}$.*

Proof. Without loss of generality we assume $\phi(y) < \phi(x)$, thus we have $r_y \leq r_x$ and $Z_{\phi(y)+1} \subseteq Z_{\phi(x)}$ and so $y \in Z_{\phi(x)}$. On the other hand, due to $x \in Z$ we must get $Z_{\phi(x)} \cap (1 + \alpha)B_{\phi(x)} = \emptyset$. Thus we obtain $d(x, y) > (1 + \alpha)r_x = (1 + \alpha)max\{r_x, r_y\}$. □

For any point x and any non-empty subset $Y \subseteq U$, let

$$charge(x, Y) = d(x, Y) + \sum\nolimits_{y \in Y} max\{0, r_y - d(x, y)\}$$

Lemma 3 ([11]). *For any non-empty subset $Y \subseteq U$, we have*

$$cost(Y) = \sum\nolimits_{x \in U} charge(x, Y) \cdot w(x)$$

Lemma 4. *Let $x \in U$ be a point, let Y be a non-empty subset of U, and let y belong to Y. If $d(x, y) = d(x, Y)$ then $charge(x, Y) \geq max\{r_y, d(x, y)\}$*

Proof. If $x \notin A_y$, then $d(x, y) > r_y$, we have $charge(x, Y) \geq d(x, y) > r_y$. Otherwise, we have $d(x, y) \leq r_y$, then we have $charge(x, Y) \geq d(x, y) + (r_y - d(x, y)) = r_y \geq d(x, y)$. The lemma is proven. □

Lemma 5. *Let $x \in U$ and $z \in Z$. If $x \in A_z$, then $charge(x, Z) \leq r_z$.*

Proof. Firstly assuming $x \in Z$, by $z \in Z$ and Lemma 2, we have $d(x, z) \geq (1 + \alpha)r_z$, which is contradicted with $x \in A_z$, so $x \notin Z$.

Now we prove $\forall y \in Z$, $y \neq z$, $d(y, x) > r_y$. Assuming this is not true, that is, $\exists y^*$ such that $y* \neq z$ and $d(y^*, x) \leq r_{y^*}$, so

$$(1 + \alpha)max\{r_{y^*}, r_z\} \leq d(y^*, z) \leq d(y^*, x) + d(x, z) \leq r_{y^*} + r_z$$

this is contradicted by $\alpha \geq 1$.

According to discussion above, we have that

$$\begin{aligned} charge(x, Z) &= d(x, Z) + (r_z - d(z, x)) + \sum_{y \in Z, y \neq z} max\{0, r_y - d(y, x)\} \\ &= d(x, Z) + (r_z - d(x, z)) \leq d(x, z) + (r_z - d(x, z)) \leq r_z \end{aligned}$$

where $d(x, z) \leq r_z$ by x belongs to A_z and the third inequality by $z \in Z$. □

Lemma 6. *Let $x \in U$ and $z \in Z$. If $x \notin A_z$, then $charge(x, Z) \leq d(x, z)$.*

Proof. If $\exists y \in Z$ such that $x \in A_y$, that is, $d(x, y) \leq r_y$. By Lemma 2 and Lemma 5, we have $d(y, z) \geq (1 + \alpha)max\{r_y, r_z\}$ and $charge(x, Z) \leq r_y$, respectively. Then we have $d(x, z) \geq d(y, z) - d(x, y) > (1 + \alpha)r_y - r_y = \alpha r_y \geq \alpha \cdot charge(x, Z) \geq charge(x, Z)$, that is, $charge(x, Z) \leq d(x, z)$.

Otherwise, if $\forall y \in Z$, we have $x \notin A_y$, by the definition of $charge(x, Z)$ and $z \in Z$ we have $charge(x, Z) = d(x, Z) \leq d(x, z)$. □

Lemma 7. *For any point $x \in U$ and non-empty subset Y, $charge(x, Z) \leq (2 + \alpha)charge(x, Y)$.*

Proof. Let y be some point in Y such that $d(x, y) = d(x, Y)$. By Lemma 1, there exists a point $z \in Z$ such that $\phi(z) \leq \phi(y)$ and $d(y, z) \leq (1 + \alpha)r_y$.

Now if $x \in A_z$, then $charge(x, Z) \leq r_z$ by Lemma 5. Thus by $\phi(z) \leq \phi(y)$, we have $r_z \leq r_y$, and since Lemma 4 implies $charge(x, Y) \geq r_y$, we obtain $charge(x, Z) \leq charge(x, Y)$.

However if $x \notin A_z$, then $charge(x, Z) \leq d(x, z)$ by Lemma 6. By triangular inequality we have $charge(x, Z) \leq d(x, y) + d(y, z) \leq [d(x, y) + (1 + \alpha)r_y]$. Moreover, by Lemma 4, we have $charge(x, Y) \geq max\{r_y, d(x, y)\}$. Then we have

$$\frac{charge(x, Z)}{charge(x, Y)} \leq \frac{d(x, y) + (1 + \alpha)r_y}{max\{r_y, d(x, y)\}} \leq 2 + \alpha$$

and the lemma is proven. □

With above lemma 3 and lemma 7, we can then clinch the following main result.

Theorem 1. *For any non-empty subset Y of U, we have*

$$cost(Z) \leq (2 + \alpha)cost(Y)$$

that is, the approximation ratio of our algorithm is $(2 + \alpha)$.

The best approximation ratio of our algorithm, which is 3 for $\alpha = 1$, is equal to the approximation ratio of Mettu and Plaxton. Though our best ratio is not less than theirs, our algorithm can present a permutation of facilities with the property that no facility can be removed as the number of facilities increases by varying the parameter α, and it can be easily used in practice since practical manager can choose the proper number of facilities by the proper α. Moreover, the result of our algorithm presents an online fashion for the different α and the different number of facilities. This property can give the practical manager much room to freely add a new facility or to delete a facility from the exist facilities.

4 Summary and Future Work

For more practical considerations, this paper presents two primary constraints on input facilities: the total number of facilities to be placed is not known in advance and a facility cannot be removed once it is placed. We gave a new variant of classic facility location problem, the strength facility location problem, and presented a group of constant-factor approximation algorithms. These algorithms, which all take quadratic time in the worst case, produce the solutions such that each result is the subset of another one. Thus the permutation of facilities obtained by all of the results present the property that no point is removed when the number of facility increases.

There are still various open problems for the future research. For example, it is interesting to design and analyze some algorithms to improve the approximation

ratio. Actually, according to the proof of Theorem 1, the condition $\alpha \geq 1$ is the main bottleneck for improving the approximation ratio. So how to relax this condition may be one of the possible directions.

Acknowledgements

This research is supported by NSF of China under Grants 10371094 and 70471035. The authors would like to thank the anonymous referees for their valuable comments.

References

1. P. K. Agarwal and M. Sharir, Efficient Algorithms for Geometric Optimization. *ACM Computing Surveys*, 30(4), pp. 412-458, 1998.
2. F.Chudak. Improved algorithms for uncapacitated facility location problem. *Proc. 5th Conference on Integer Programming and Combinatorial Optimization*, LNCS 1412, pp.180-194,1998.
3. J. Current, M. Daskin and D. Schilling, Discrete network location models. In Z. Drezner and H. W. Hammacher, editors, *Facility Location: Applications and Theory*, Springer, pp.81-118, 2002.
4. J. Current, S. Ratick and C.Revelle, Dynamic facility location when the total number of facilities is uncertain: a decision analysis approach, *European Journal of Operation Research*, 110(3), pp. 597-609, 1998.
5. M. Charikar and S. Guha. Improved combinatorial algorithms for facility location and $k-$median problems, *Proc. FOCS'99*, pp.378-388, 1999.
6. F. Chudak and D. B. Shmoys. Improved approximation algorithms for capacitated facility location problem. *Proc. 10th ACM-SIAM Symposium on Discrete Algorithms*, 1999.
7. R. Fleischer, M. J. Golin and Z. Yan, Online Maintenance of $k-$medians and $k-$covers on a line, *Proc. 9th Scandinavian Workshop on Algorithm Theory*, LNCS 3111, pp.102-113, 2004.
8. S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms, *Proc. of the 9th ACM-SIAM Symposium on Discrete Algorithms*,1998, Also in *Journal of Algorithms*, 31, pp. 228-248,1999.
9. K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and $k-$median problems. *Proc. FOCS'99*, 1999.
10. M. Mahdian, Y. Ye and J. Zhang, Improved approximation algorithms for metric facility location problems, *Proc. of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX'02)*, pp.229-242, 2002.
11. R. R. Mettu and C. G. Plaxton. The online median problem. *Proc. FOCS'00*, pp. 339-348, 2000.
12. C.G.Plaxton. Approximation algorithms for hierarchical location problems. *Proc. STOC'03*, pp. 40-49,2003.
13. Z. Qin, Y. Xu and B. Zhu, On some optimization problems in obnoxious facility location, *Proc. COCOON'00*, LNCS 1858, pp.320-329, 2000.

14. D. B. Shmoys, Approximation algorithms for facility location problems. In K. Jansen and S. Khuller, editors, *Approximation Algorithms for Combinatorial Optimization*, LNCS 1913, pp.27-33, 2000
15. D. B. Shmoys, E. Tardos and K. Aardal. Approximation algorithms for facility location problems, *Proc. STOC'97*, pp. 265-274,1997
16. M. Sviridenko. An 1.528-approximation algorithm for the metric uncapacitated facility location problem. *Proc. of the 9th Conference on Integer Programming and Combinatorial Optimization*, 2002.