Non-Metric Multicommodity and Multilevel Facility Location

Rudolf Fleischer⋆, Jian Li, Shijun Tian, and Hong Zhu⋆⋆

Department of Computer Science and Engineering
Shanghai Key Laboratory of Intelligent Information Processing
Fudan University
Shanghai 200433, China
[rudolf,lijian83,tiansj,hzhu]@fudan.edu.cn

Abstract. We give logarithmic approximation algorithms for the non-metric uncapacitated multicommodity and multilevel facility location problems. The former algorithms are optimal up to a constant factor, the latter algorithm is far away from the lower bound, but it is the first algorithm to solve the general multilevel problem. To solve the multicommodity problem, we also define a new problem, the friendly tour operator problem, which we approximate by a greedy algorithm.

1 Introduction

The facility location problem and many variants have been studied extensively in both the operation research and computer science niterature [11, 13, 17]. In the basic *uncapacitated facility location problem (*`UFL`*)* we are given a set $\mathcal{C}$ of n *clients* and a set $\mathcal{F}$ of m *facilities*. Each facility $f \in \mathcal{F}$ has an opening cost f^0, and connecting a client $c \in \mathcal{C}$ to f costs c^f. These costs can be arbitrary real numbers, although they will be positive in most applications. In this paper, all facility location problems will be uncapacitated, so we will henceforth omit 'uncapacitated' when speaking about facility location problems.

We may consider the sets $\mathcal{C}$ and $\mathcal{F}$ as the two sides of a bipartite graph. Consider a set E of edges (or *links*) between $\mathcal{C}$ and $\mathcal{F}$. Let F_E be the subset of facilities incident to at least one edge. If (c, f) is an edge in E, then we say c can *satisfy* its demand, and f *satisfies* the demand of c. E is a feasible solution if every client in $\mathcal{C}$ can satisfy its demand (i.e., every client is incident to at least one edge). The cost of E is defined as

$$cost(E) = \sum_{f \in F_E} f^0 + \sum_{(c,f) \in E} c^f \ ,$$

⋆ The work described in this paper was partially supported by the National Natural Science Fund China (grants #60496321, #60373021, and #60573025) and the Shanghai Science and Technology Development Fund (grant #03JC14014).

⋆⋆ The order of authors follows the international standard of alphabetic order of the last name. In China, where first-authorship is the only important aspect of a publication, the order of authors should be Jian Li, Shijun Tian, Rudolf Fleischer, and Hong Zhu.

where the first sum is the *startup cost* of F_E and the second sum is the *connection cost* (or *link cost*). UFL is the problem of finding a solution of minimum cost.

In *metric* UFL, the link costs obey the triangle inequality. In particular, in *geometric* UFL the clients and facilities are points in the plane (or more general, in $\mathbb{R}^d$).

In this paper we will discuss four variants of non-metric UFL, two variants of the multicommodity facility location problem, the multilevel facility location problem,a and the multilevel concentrator location problem. These problems are defined in Section 2. In Section 3 we state our new results and review previous related work. In Section 4 we give our new asymptotically optimal approximation algorithms for the two multicommodity facility location variants. In Section 5, we then give approximation algorithms for the multilevel facility location problem and for the multilevel concentrator location problem. We end the paper with some remarks and open problems in Section 6.

2 The Models

2.1 Multicommodity Facility Location

The *multicommodity facility location problem (*MCFL*)* generalizes UFL by introducing a set $\mathcal{S}$ of k different *commodities* (or *services*). In UFL, we only have a single commodity. Each client c *demands* one unit of each commodity in a subset $S_c \subseteq \mathcal{S}$, whereas each facility f can only offer a subset $S_f \subseteq \mathcal{S}$ of commodities (but arbitrary many units of each type). A collection of links E is a feasible solution if each client can satisfy its demand for each of its commodities. In a more general setting, we might also consider a weighted version of MCFL where clients have a certain non-negative demand for each commodity and facilities have only limited capacity for each demand.

Note that the link costs do not scale with the number of commodities served by the link. Once established, a link can be used to satisfy the demands for several commodities without additional cost. MCFL is a natural model, for example, for planning the locations of network switches (for a computer network in a large building, or telephone switchboards in a city) where we want to minimize the setup cost plus the cost of connecting each client to a switch.

We could generalize MCFL by charging an independent link cost for the commodities, i.e., if a client satisfies his demands for several commodities from one facility, it must pay the link cost for each commodity. However, this problem can be reduced to UFL by splitting each client into several clients at the same location, one for each commodity.

Another generalization of UFL is the *facility location with service installation cost problem (*FLSC*)* [14]. If facility f satisfies the demand for commodity s of some client, it must pay a one-time *installation cost* f^s for this commodity. Note that now a feasible solution must specify the links E and for each facility f the set $D_f \subseteq S_f$ of commodities provided by f, and each client must be able to satisfy its demands from some facilities that provide the commodities and have paid the respecitve startup costs. The startup cost of f is then

$$f^0 + \sum_{s \in D_f} f^s \, .$$

Ravi and Sinha called these cost functions *linear* [12].

2.2 Multilevel Facility Location

Let $k \geq 1$ be some integer. In the *k-level facility location problem (k-*`UFL`*)* we consider a $(k+1)$-layer graph, where the first layer $\mathcal{C} = \mathcal{F}_0$ is the set of clients, and the next k layers $\mathcal{F}_1, \ldots, \mathcal{F}_k$ are sets of facilities. Each edge (link) has a cost. We are interested in paths connecting a client in layer $\mathcal{F}_0$ with some facility in the last layer $\mathcal{F}_k$. The cost of such a path, call a *link path*, is the sum of its individual link costs. A feasible solution is a set E of link paths such that each node in $\mathcal{F}_0$ is incident to at least one link path. Intuitively, each client has a demand for a commodity available at any facility in layer k, which then must be routed to the client via facilities at the intermediate $k-1$ layers. Note that in this model an edge can incur multiple cost if it is used in several link paths.

If an edge only incurs cost once even if it is shared by several link paths, we are dealing with the *k-level concentrator location problem (k-*`LCLP`*)*. Here, each client must be satisfied by a facility in $\mathcal{F}_1$, each facility in $\mathcal{F}_1$ must be satisfied by a facility in $\mathcal{F}_2$, etc. Formally, our goal is to choose subsets $\emptyset \neq V_t \subseteq \mathcal{F}_t$, for $1 \leq t \leq k$, such that

$$\sum_{j \in D} \min_{k \in V_1} c_{jk} + \sum_{t=1}^{k-1} \sum_{j \in V_t} \min_{i \in V_{t+1}} c_{ji} + \sum_{t=1}^{k} \sum_{i_t \in V_t} f_{i_t}$$

is minimized.

3 Background and New Results

3.1 Multicommodity Facility Location

Facility location problems are usually NP-hard, and approximation algorithms for many variants have been studied [11, 13, 17]. The multicommodity facility location problem, however, has only been studied recently. Ravi and Sinha [12] gave a first $O(\log |\mathcal{S}|)$-approximation algorithm for metric `UFL` when each client can only demand a single commodity. The result generalizes to the case of clients demanding several commodities, but if they satisfy them over the same link, the link cost will also be charged several times (so this model is different from `MCFL`). Their result is based on an IP formulation of the problem that can be approximated by rounding fractional LP solutions.

Shmoys *et al.* [14] gave a primal-dual 6-approximation algorithm for `FLSC` under the assumption that facilities can be ordered by increasing installation costs, with the same order for all commodities.

We present in this paper the first approximation algorithms for non-metric MCFL and FLSC. They are purely combinatorial, not based on IP-approximations. For MCFL we give an H_h-approximation, where h is the total number of commodities demanded by all the clients, i.e., $h = \sum_{c \in \mathcal{S}} |S_c|$. Since $h \leq nk$, this is an $O(\log(nk))$-approximation. For FLSC we give a $(3H_h)$-approximation, which is also an $O(\log(nk))$-approximation. If all facilities f have startup-cost $f^0 = 0$, the approximation ratio is only $2H_h$. We also show that our approximation ratios are asymptotically optimal. This follows easily from the non-approximability lower bound bound for the set cover problem by Feige [5].

Both algorithms are based on the well-known greedy *minimum weight set cover (*SC*)* approximation algorithm by Chvátal [4]. This algorithm iteratively picks the set for which the ratio of weight over newly covered elements is minimized, giving a SC approximation of ratio of H_d, where d is the number of elements in the set to be covered. As Hochbaum observed [8], the same algorithm, with the same approximation factor, can be applied to other problems as long as they can be reduced to SC and as long as it is possible to compute in every step in polynomial time the subset (or its equivalent structure) minimizing the relative weight of the newly covered elements. For UFL, this condition is fulfilled, with $\mathcal{C}$ the set to be covered, so there is an H_n-approximation for UFL [8].

We will see in Section 4 that we can also easily use the SC approximation for MCFL. However, for FLSC computing the minimum relative weight set in every step is rather difficult. Since we cannot easily compute an optimal set, we use a 3-approximation, which is the reason for the factor of 3 in the $3H_n$-approximation ratio of the FLSC algorithm. The 3-approximation is the solution of a new problem we define, the *friendly tour operator problem (*FTO*)*. Such a quasi-greedy approach has been used before, see for example [7].

3.2 Multilevel Facility Location

Multilevel facility location has a long history in operations research [1, 3, 9, 16]. Of course, 1-UFL is nothing but UFL. Shmoys *et al.* [15] gave the first constant factor approximation algorithm for the metric case, and the current best known result is a 1.52-approximation by Mahdian *et al.* [10]. Guha and Khuller showed that it is unlikely to be approximated within a factor of 1.463 [6].

Shmoys *et al.* [15] extended their filtering and rounding technique for metric 1-UFL to metric 2-UFL, resulting in a 3.16-approximation algorithm. Later, Aardal *et al.* [2] showed that metric k-UFL can be approximated in polynomial time by a factor of 3 for any positive integer k using a linear programming relaxation. For small values of k, better approximation algorithms are known: 1.77 for $k = 2$, 2.51 for $k = 3$, and 2.81 for $k = 4$ [18]. For non-metric 2-UFL, Zhang gave an $O(\ln n)$-approximation [18].

In this paper, we present the first approximation algorithm for general non-metric k-UFL. The approximation ratio of our algorithm is $O(\ln^k n)$. The algorithm is defined inductively, starting with the classical $O(\ln n)$-approximation for 1-UFL. In the inductive step, we again make use of the greedy SC approximation technique. With a very similar algorithm,we can also solve the k-level concentrator location problem where we just have a hierarchy of k levels of facilities.

4 Multicommodity Facility Location

4.1 Set Cover and Facility Location

We first quickly review the relationship between UFL and SC, since this is at the heart of all our algorithms. We follow loosely the exposition by Vygen [17, Section 3.1].

In the set cover problem we are given a finite set U, a family $\mathcal{X}$ of subsets of U which together cover U, and non-negative weights $c(V)$ on the sets $V \in \mathcal{X}$. The task is to find a subset $\mathcal{Y} \subseteq \mathcal{X}$ covering U of minimum total weight. SC is a special case of UFL: let the elements in U be the clients, the subsets in $\mathcal{X}$ the facilities, the weight of a set $V \in \mathcal{X}$ the startup cost of the facility, and let the link cost of client $c \in U$ to facility $V \in \mathcal{X}$ be zero if $c \in V$ and infinity if $c \notin V$. Now, every solution to UFL corresponds to a set cover of the same cost, and vice versa.

Conversely, UFL can be considered a special case of set cover. For an instance of UFL, define a *star* to be a pair (f, C) with $f \in \mathcal{F}$ and $C \subseteq \mathcal{C}$ (meaning that we link all the clients in C to facility f). The *cost* of this star is

$$f^0 + \sum_{c \in C} c^f ,$$

and its *effectiveness* is

$$\frac{f^0 + \sum_{c \in C} c^f}{|C|} ,$$

i.e., the relative cost per client in the star. Then we can define a SC instance by choosing $\mathcal{C}$ as the set U and all possible subsets of $\mathcal{C}$ as $\mathcal{X}$, where $C \subseteq \mathcal{C}$ has cost equal to the minimum cost of a star (f, C), minimized over all $f \in \mathcal{F}$. Now, an optimal solution to SC corresponds to an optimal solution to UFL of the same cost, and vice versa.

Chvátal's greedy SC approximation algorithm iteratively picks a set for which the ratio of weight over newly covered elements is minimized [4]. If we apply this algorithm to UFL, we must in every step pick the most effective star. Although there are exponentially many stars, we do not need to compute them all. Instead, we can find the most effective star among the stars (f, C_k^f), where f is an arbitrary facility, and C_k^f denotes the first k clients in a linear order with nondecreasing link cost to f, for $k = \{1, \ldots, n\}$. Having identified the most effective star, we then open the facility and henceforth disregard all clients in this star. We refer to this algorithm as the *standard star algorithm.*

4.2 Approximating `MCFL`

In `MCFL`, each client can demand several commodities. If the client decides to satisfy its demand for one commodity from a facility, then it can, without additional cost, satisfy all demands that the facility provides from that facility. This means, if we pick a star in the standard star algorithm, the facility should satisfy all unsatisfied demands of the clients in the star. Thus, we should change the definition of effectiveness of a star (f, C) to

$$\frac{f^0 + \sum_{c \in C} c^f}{\sum_{c \in C} |S_f \cap S_c|} .$$

In the definition of C_k^f we now order the clients in linear order with nondecreasing link cost divided by number of demands that could maximally be satisfied by f, i.e., we sort them by nondecreasing

$$\frac{c^f}{|S_f \cap S_c|} .$$

Theorem 1. *The modified standard star algorithm gives an H_h-approximation, where h is the total number of commodities demanded by all the clients.*

Proof. We have to show that the modified linear order of clients in the definition of C_k^f guarantees that we indeed find a most effective star. This proof is straightforward and omitted in this extended abstract. □

4.3 The Friendly Tour Operator Problem

To solve `FLSC`, we must define a new problem that we need as a subroutine, the *friendly tour operator problem (*`FTO`*)*. Consider a tour operator who would like to organize a tour for tourists. Each tour t incurs a fixed cost t^0 (maybe the profit of the tour operator). There is a certain finite set $\mathcal{A}$ of actions that can be arbitrarily combined in a tour. Let A_t denote the set of actions offered in tour t. Each action a incurs a cost t^a (maybe an entrance fee). There is also a set $\mathcal{T}$ of tourists. Each tourist x demands to participate in some set A_x of actions. He will only join the tour t if $A_x \subseteq A_t$. The total cost of t will be

$$t^0 + \sum_{a \in A_t} t^a ,$$

which is equally shared by all participants. The goal of the friendly tour operator is not to maximize his profit, but to offer a tour of minimum cost for the participants.

We could model the problem as a hypergraph problem, where the nodes are the actions and the hyperedges are the tourists. Then the problem generalizes the densest subgraph problem which is NP-hard. So we cannot solve `FTO` optimally in polynomial time. But we can find a good approximation to the best tour by a simple greedy algorithm, `Approx-FTO`.

Starting with all actions, in each step we first compute the average cost of the current action set and then discard that action (and all tourists demanding it) that maximizes the quotient of the cost of the action and the number of tourists demanding the action (i.e., intuitively we discard an action if it has high cost and is not high in demand). In the sequence of action sets computed, we then choose the one with lowest average cost.

Theorem 2. *Let d be the maximum number of actions any tourist demands. Then,* `Approx-FTO` *achieves an approximation factor of d if $t^0 = 0$ and a factor of $d+1$ if $t^0 \geq 0$.*

Proof. Let $A^\star$ be an optimal set of actions and OPT be the value of the optimal solution. Let $T^\star$ be the number of tourists participating in the optimal tour, and let $D_b^\star$ denote how many of them are demanding action $b \in A^\star$.

Let a be the first action in $A^\star$ deleted by `Approx-FTO`. Right before this happens, let A be the current set of actions, T be the number of remaining tourists, *cost* be the current average cost, and for any $b \in A$ let D_b denote the number of tourists demanding action b. Clearly, $D_b^\star \leq D_b$ for all b. Therefore, $\frac{t^a}{D_a} \leq OPT$, because otherwise $A^\star - \{a\}$ would be a better solution than $A^\star$.

We choose a in the next step because $\frac{t^a}{D_a} \geq \frac{t^b}{D_b}$, for all $b \in A$. Since each tourist can demand at most d actions, we have $\sum_{b\in A} D_b \leq d \cdot T$. Putting all together, we obtain

$$cost \leq \frac{t^0 + \sum_{b\in A} t^b}{T} \leq \frac{t^0}{T^\star} + \frac{d \cdot \sum_{b\in A} t^b}{\sum_{b\in A} D_b} \leq OPT + d \cdot \frac{t^a}{D_a} \leq (d+1)\cdot OPT\,.$$

If $t^0 = 0$, the first OPT term vanishes and we get a d-approximation. □

As the following example shows, our analysis of `Approx-FTO` is tight if $d = 2$. In this case, we can model `FTO` as a graph problem with actions as nodes and edges as tourists. We assume startup cost $t^0 = 0$. Consider the graph G which is the union of $K_{n,n}$ and S_{2n}, where $K_{n,n}$ is the complete bipartite graph with node partitions U and V, where $|U| = |V| = n$, and S_{2n} is a star with $2n+1$ nodes, namely a center node v and $2n$ leaves. Each node in U has cost $1+\epsilon$, where $\epsilon > 0$ is sufficiently small. The cost of v is 2, while the leaves all have cost $\frac{1}{n}$. The nodes in V have cost zero. The optimal solution is in this case the $K_{n,n}$, with minimum average cost $\frac{1+\epsilon}{n}$. But `Approx-FTO` will first delete a node in $K_{n,n}$ (which has maximum ratio $\frac{1+\epsilon}{n}$) and eventually find S_{2n} as the solution with average cost $\frac{2+2n\cdot\frac{1}{n}}{2n} = \frac{2}{n}$.

4.4 Approximating `FLSC`

In `FLSC`, each facility has some additional startup cost for providing a commodity. Therefore, it may now happen that a client satisfies one demand from one facility but a second demand from another facility although the first facility could also satisfy the second demand (but its startup cost for this demand is too high).

We must redefine cost and effectiveness of a star, and even stars itself. Consider a facility f at some step of the algorithm. If it had been used before, its startup cost is now zero. If some of its commodities are already in use from earlier clients, their startup costs are also zero. A star is now a triple (f, C, S), where S is a subset of commodities provided by the star. We may assume that S always includes all commodities that are already in use at the facility (they can now be used for free by other clients). The cost of the star is then defined as

$$f^0 + \sum_{s \in S} f^s + \sum_{c \in C} c^f \ ,$$

and its *effectiveness* is

$$\frac{f^0 + \sum_{s \in S} f^s + \sum_{c \in C} c^f}{\sum_{c \in C} |S \cap S_c| \ .}$$

After choosing a most effective star, we only discard the demands of the clients that have been satisfied (a client can be discarded when all its demands are satisfied).

The problem is how to find a most effective star in polynomial time. There does not seem to be a natural linear order of clients in the definition of C_k^f that guarantees that we indeed find the most effective star among the C_k^f. Since we cannot find the best star, we approximate it. Note that to compute a most effective star we only have to solve an `FTO` for each facility f and then choose the cheapest of all of them. To be more precise, for fixed f, the `FTO` uses $t^0 = f^0$. There are $n + k$ actions, one for each commodity and one for each link from f to a client. The cost of an action is the corresponding cost in `FLSC`. For each client $c \in \mathcal{C}$ and unsatisfied comodity $s \in S_c$, there is a tourist demanding the two actions c and the link from f to c.

Theorem 3. *The modified standard star algorithm using* `Approx-FTO` *as a subroutine to approximate a most efficient star gives a $2H_h$-approximation, where h is the total number of commodities demanded by all the clients, if $f^0 = 0$ for all $f \in \mathcal{F}$, and a $3H_h$-approximation in the general case.*

Proof. The theorem follows from the standard star algorithm together with the approximation of the most effective star given in Theorem 2. □

4.5 Lower Bounds

`FLSC` is clearly a generalization of `MCFL`, so any lower bound for the approximation factor of `MCFL` is also a lower bound for `FLSC`.

Theorem 4. *There is no polynomial approximation algorithm for* `MCFL` *and* `FLSC` *with an approximation factor of $(1 - \epsilon) \cdot max\{\ln n, \ln k\}$, for any $\epsilon > 0$, where n is the number of clients and k is the number of commodities.*

Proof. We give two reductions from SC. Let $|U| = n$. Recall that there is no polynomial time approximation algorithm for SC with an approximation factor of $(1 - \epsilon) \cdot \ln n$, for any $\epsilon > 0$, unless $NP \subseteq DTIME[n^{O(\log \log n)}]$ [5].

The reduction given in Subsection 4.1, where we have a single commodity and clients correspond to elements in U, gives a lower bound of $\ln n$.

In the second reduction, let each commodity correspond to a unique element in U. There is only one client demanding all commodities. For each subset in $\mathcal{X}$, there is a facility with startup cost 1 providing the corresponding commodities. All connection costs are zero. Now any set cover corresponds to a MCFL solution of the same cost. Thus, we cannot approximate MCFL with a factor better than $\ln k$. □

5 k-Level Facility Location

We must define a more general version of k-UFL, k-UFL$_\ell$, which has an additional input parameter ℓ. In this problem, we can first choose s subset of ℓ clients which is then optimally served by some set of facilities. Note that k-UFL$_m$ is just the original k-UFL.

We define our approximation algorithm for k-UFL inductively. First, we give an $O(\ln \ell)$-approximation algorithm for 1-UFL$_\ell$. Then we show how to lift an $O(\ln^{k-1} \ell)$-approximation for $(k-1)$-UFL$_\ell$ up to an $O(\ln^k \ell)$-approximation for k-UFL$_\ell$.

5.1 Approximating 1-UFL$_\ell$

The $\ln \ell$-approximation algorithm for 1-UFL$_\ell$ is very similar to the greedy algorithm for 1-UFL. When we compute the most effective star for facility f, we only consider sets C_k^f for $k = 1, \dots, \ell$, and we stop when we have satisfied ℓ clients.

Theorem 5. *The modified standard star algorithm computes a $\ln \ell$-approximation for 1-UFL$_\ell$, for any $1 \leq \ell \leq m$.* □

5.2 Approximating k-UFL$_\ell$

Suppose we have an approximation algorithm APPROX-$(k-1)$-UFL$_\ell$ for $(k-1)$-UFL$_\ell$ for every $1 \leq \ell \leq n$. Then we can construct an algorithm APPROX-k-UFL$_\ell$ for k-UFL$_\ell$ as follows.

Consider a fixed facility $f \in \mathcal{F}_k$. We construct an instance for $(k-1)$-UFL$_\ell$ as follows. The set of clients remains unchanged, also the set of facility levels $\mathcal{F}_1, \dots, \mathcal{F}_{k-1}$. What changes is the connection cost between $\mathcal{F}_{k-2}$ and $\mathcal{F}_{k-1}$. We increase the cost of each original edge (u, v) between the two levels by the cost of the original edge (v, f). Intuitively, we are extending the last edge on a path from a client to a node in level $k-1$ by the edge leading to f in level k.

In the standard star algorithm, we would now compute, for each facility, the best way to connect it with $1, 2, 3, \dots$ clients, and then choose the cheapest star. Here we cannot easily compute these values. Instead, we again approximate them.

Let $cost(f,j)$ be the cost of an approximation computed by APPROX-$(k-1)$-UFL$_j$, for $1 \leq j \leq \ell$. We compute all these values for all f and j and determine the smallest one. This tells us which facility f in level k to choose. We choose all the facilities and connections computed in the corresponding approximation of the $(k-1)$-level problem, and we connect f to all facilities chosen on level $k-1$.

Theorem 6. *If* APPROX-$(k-1)$-UFL$_\ell$ *can achieve an approximation factor of* $O(\ln^{k-1} \ell)$*, for all* $1 \leq \ell \leq n$*, then* APPROX-k-UFL$_\ell$ *computes an* $O(\ln^k \ell)$*-approximation.* □

Theorem 7. *There exists a* $\ln^k n$*-approximation algorithm for* k-UFL*.* □

5.3 The *k*-Level Concentrator Location Problem

It is not hard to modify our algorithm for k-UFL to approximate k-LCLP. The only change is in the inductive step when we change the connection costs of edges between layers $k-2$ and k. Instead, we now increase the startup costs of facilities on layer $k-1$ by the cost of the edge to facility f on layer k.

Theorem 8. *There exists a* $\ln^k n$*-approximation for* k-LCLP*.* □

6 Conclusions

We presented the first logarithmic approximation algorithms for the non-metric multicommodity facility location problem. Note that in our model the connection costs do not scale with the number of commodities that use a connection. This actually generalizes the case where connection costs scale. For FLSC, our algorithms have an additional constant factor of 2 or 3, which may not be necessary for an optimal approximation algorithm.

We also presented the first poly-logarithmic approximation algorithm for the non-metric k-level facility location problem. We conjecture that this problem admits a logarithmic approximation for any $k \geq 1$.

Acknowledgements

We would like to thank the reviewers for reading our hastily prepared last-minute submission, for which we apologize. Their comments helped us to improve the presentation of our results.

References

1. K. Aardal. *On the solution of one and two-level capacitated facility location problems by the cutting plane approach.* Ph.D. thesis, Université Catholique de Louvain, Louvain-la-Neuve, Belgium, 1992.
2. K. Aardal, F. A. Chudak, and D. B. Shmoys. A 3-approximation algorithm for the k-level uncapacitated facility location problem. *Information Processing Letters*, 72(5-6):161–167, 1999.
3. K. Aardal, M. Labbé, J. Leung, and M. Queyranne. On the two-level uncapacitated facility location problem. *INFORMS Journal on Computing*, 8:289–301, 1996.
4. V. Chvátal. A greedy heuristic for the set cover problem. *Mathematics of Operations Research*, 4:233–235, 1979.
5. U. Feige. A threshold of $\ln n$ for approximating set-cover. *Journal of the ACM*, 45(4):634–652, 1998.
6. S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31:228–248, 1999.
7. S. Guha and S. Khuller. Improved methods for approximating node weighted Steiner trees and connected dominating sets. *Information and Computation*, 150:57–74, 1999.
8. D. S. Hochbaum. Heuritics for the fixed cost median problem. *Mathematical Programming*, 22(2):148–162, 1982.
9. L. Kaufmann, M. vanden Eede, and P. Hansen. A plant and warehouse location problem. *Operational Research Quaterly*, 28:547–557, 1977.
10. M. Mahdian, Y. Ye, and J. Zhang. A 1.52-approximation algorithm for the uncapacitated facility location problem. In *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX'02).* Springer Lecture Notes in Computer Science 2462, pages 127–137, 2002.
11. P. Mirchandani and R. Francis, editors. *Discrete Location Theory.* John Wiley & Sons, Inc., Chichester, 1990.
12. R. Ravi and A. Sinha. Multicommodity facility location. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 342–349, 2004.
13. D. B. Shmoys. Approximation algorithms for facility location problems. In *Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX'00).* Springer Lecture Notes in Computer Science 1913, pages 27–33, 2000.
14. D. B. Shmoys, C. Swamy, and R. Levi. Facility location with service installation costs. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 1088–1097, 2004.
15. D. B. Shmoys, E. Tardos, and K. I. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the 29th ACM Symposium on the Theory of Computation (STOC'97)*, pages 265–274, 1997.
16. D. Tcha and B. Lee. A branch-and-bound algorithm for the multi-level uncapacitated location problem. *European Journal on Operations Research*, 18:35–43, 1984.
17. J. Vygen. Approximation Algorithms for Facility Location Problems (Lecture Notes). Technical Report Report No. 05950-OR, Research Institute for Discrete Mathematics, University of Bonn, 2005.

18. J. Zhang. Approximating the two-level facility location problem via a quasi-greedy approach. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 808–817, 2004.