Improved Combinatorial Approximation Algorithms for the k-Level Facility Location Problem

Alexander Ageev* Yinyu Ye† Jiawei Zhang ‡

Abstract

In this paper we present improved combinatorial approximation algorithms for the k-level facility location problem. First, by modifying the path reduction developed in [2], we obtain a combinatorial algorithm with a performance factor of 3.27 for any $k \geq 2$, thus improving the previous bound of 4.56 achieved by a combinatorial algorithm. Then we develop another combinatorial algorithm that has a better performance guarantee and uses the first algorithm as a subroutine. The latter algorithm can be recursively implemented and achieves a guarantee factor $h(k)$, where $h(k)$ is strictly less than 3.27 for any k and tends to 3.27 as k goes to ∞. The values of $h(k)$ can be easily computed with an arbitrary accuracy: $h(2) \approx 2.4211$, $h(3) \approx 2.8446$, $h(4) \approx 3.0565$, $h(5) \approx 3.1678$ and so on. Thus, for the cases of $k = 2$ and $k = 3$ the second combinatorial algorithm ensures an approximation factor substantially better than 3, which is currently the best approximation ratio for the k-level problem provided by the non-combinatorial algorithm due to Aardal, Chudak, and Shmoys [1].

*Sobolev Institute of Mathematics, Novosibirsk, Russia. Email: ageev@math.nsc.ru. Research was partially supported by the Russian Foundation for Basic Research, project codes 01-01-00786, 02-01-01153, by INTAS, project code 00-217, and by the Programme "Universities of Russia", project code UR.04.01.012.

†Department of Management Science and Engineering, Stanford University, Stanford, CA 94305, USA. Email: yinyu-ye@stanford.edu. Research supported in part by NSF grant DMI-0231600.

‡Department of Management Science and Engineering, Stanford University, Stanford, CA 94305, USA. Email: jiazhang@stanford.edu. Research supported in part by NSF grant DMI-0231600.

1 Introduction

In the k-level facility location problem (for brevity, k-LFLP) we are given a complete $(k+1)$-partite graph $G = (D \cup \mathcal{F}_1 \cup \ldots \cup \mathcal{F}_k; E)$ whose node set is the union of $k+1$ disjoint sets $D, \mathcal{F}_1, \ldots, \mathcal{F}_k$ and the edge set E consists of all edges between these sets. The nodes in D are called demand sites and the nodes in $\mathcal{F} = \mathcal{F}_1 \cup \ldots \cup \mathcal{F}_k$ are facilities (of level $1, \ldots, k$ respectively). We are given edge costs $c \in \mathbb{R}_+^E$ and opening costs $f \in \mathbb{R}_+^{\mathcal{F}}$ (i. e., opening a facility $i \in \mathcal{F}$ incurs a cost $f_i \geq 0$).

The objective is to open a set of facilities $X_t \subseteq F_t$ on each level $t = 1, \ldots, k$ and to connect each demand site $j \in D$ to a path (or chain) $\varphi(j) = (i_1(j), i_2(j), \ldots, i_k(j))$ along open facilities $i_1(j) \in X_1, i_2(j) \in X_2, \ldots, i_k(j) \in X_k$ so that the total cost of opening and connecting

$$\sum_{i \in X_1 \cup \ldots \cup X_k} f_i + \sum_{j \in D} \Big(c(j, i_1(j)) + c(i_1(j), i_2(j)) + \ldots + c(i_{k-1}(j), i_k(j)) \Big) \quad (1)$$

is minimized.

In this paper we consider the metric case of the problem where c is induced by a metric on the whole set of nodes $V = D \cup \mathcal{F}_1 \cup \ldots \cup \mathcal{F}_k$. Recent applications of metric facility location problems include finding product clustering, cost-effective placement of servers on the internet, and optimized supply-chains [6].

Since the metric k-LFLP is NP-hard, the major part of research work is concentrated on designing approximation algorithms. We say that an algorithm for a minimization problem with non-negative objective function is a *ρ-approximation algorithm* if it runs in polynomial time and for any instance, outputs a solution of cost at most ρ times the optimum.

The special case of k-LFLP where $k = 1$ (1-LFLP) is nothing but the well-known (metric) uncapacitated facility location problem (for brevity, UFLP). It is known that the existence of a 1.463-approximation algorithm for solving UFLP would imply $NP \notin DTIME[n^{O(\log \log n)}]$ [7]. In recent years quite a number of approximation algorithms have been developed for solving UFLP. The currently best approximation algorithm due to Mahdian, Ye, and Zhang [11] achieves a factor of 1.517. See Shmoys [13] and [11] for a detailed survey on approximation algorithms for UFLP.

Obviously, the lower approximability bound 1.463 also applies to k-LFLP. On the positive side, it is known that k-LFLP can be solved within a factor

of 3 by an LP rounding algorithm due to Aardal, Chudak, and Shmoys [1]. A drawback of this algorithm is that it includes a phase of solving a linear relaxation with exponential number of variables. Despite the fact that this relaxation can be solved by the ellipsoid method in polynomial time, the algorithm would be inefficient in practice. For this reason, very recently several combinatorial approximation algorithms have been developed to solve this problem. These algorithms run in strongly polynomial time but with a sacrifice in the performance guarantee. The first such algorithm by Meyerson, Munagala, and Plotkin [12] had an approximation factor of $O(\ln |\mathcal{D}|)$. A constant factor of 9.2 was later obtained by Guha, Meyerson, and Munagala [8]. Bumb and Kern [3] developed a dual ascent algorithm which had a performance guarantee of 6. Ageev [2] established that any ρ-approximation algorithm for UFLP could be translated to a 3ρ-approximation algorithm for k-LFLP. Thus, the algorithm in [11] yields a combinatorial 4.56-approximation algorithm for k-LFLP. We will refer to this approach as the *path reduction* technique. It should be noted that Edwards [4] proposed a reduction similar to that in [2] but his construction requires running time exponential in k.

None of the above algorithms has a performance guarantee better than 3. Whether or not k-LFLP can be approximated in polynomial time by a factor less than 3 has become a challenging open question in this field.

In this paper we present improved combinatorial approximation algorithms for the k-level facility location problem.

First, by modifying the path reduction of the k-level problem to the 1-level case developed in [2], we obtain a combinatorial algorithm with a performance guarantee of 3.27 for any k, thus improving the previous bound of 4.56. The algorithm runs in time $O(m_1^3 n^3 + m^2 n)$ where $m = |\mathcal{F}|$, $m_1 = |\mathcal{F}_1|$, and $n = |D|$. Note that the approximation ratio of this path reduction algorithm is fairly close to a factor of 3 provided by the LP rounding algorithm [1]. Furthermore, this theoretical result in some sense explains why in computational experiments the path reduction based algorithms perform better than the LP rounding algorithm, as it has been observed by Edwards [4].

Though the intuition suggests that k-LFLP for small values of $k \geq 2$ may be better approximable than the general problem, our path reduction algorithm, as all the previous algorithms, has the same approximation factor for each k. This drawback motivated our work on a better algorithm whose performance factor would be an increasing function of k with values strictly less than 3.27. Our efforts resulted in a recursive combinatorial algorithm for

k-LFLP, which is presented in the second part of this paper. It is based on a combination of our path reduction algorithm and a recursive reduction of k-LFLP to $(k-1)$-LFLP and UFLP. The algorithm runs in time $O(k(m_1^3n^3 + m^2n))$ and achieves an approximation factor $h(k)$, where $h(k)$ is strictly less than 3.27 for any $k \geq 1$ and tends to 3.27 as k tends to ∞. The values of $h(k)$ can be easily computed with an arbitrary accuracy. In particular, $h(2) \leq 2.4211$, $h(3) \leq 2.8446$, $h(4) \leq 3.0565$, $h(5) \leq 3.1678$. Thus, for 2-LFLP and 3-LFLP, the second algorithm achieves an approximation factor substantially better than 3.

2 The path reduction algorithm

In this section we present a parameterized version of the path reduction, which in combination with the greedy algorithm developed in [11] yields a 3.27-approximation algorithm for solving k-LFLP.

2.1 Definitions and notation

Denote by $\mathcal{P}$ the set of all paths of length $k-1$ connecting a node in $\mathcal{F}_1$ to a node in $\mathcal{F}_k$. For a path $p = (i_1, i_2, \cdots, i_k) \in \mathcal{P}$, let $c(p) = \sum_{t=2}^{k} c(i_{t-1}, i_t)$. For any subset $X \subseteq \mathcal{F}$, let $f(X) = \sum_{i \in X} f_i$ and let $\mathcal{P}(X)$ denote the subset of paths in $\mathcal{P}$ passing through facilities in X.

Let $\mathcal{M}$ be an instance of k-LFLP and SOL be a solution of it. Recall that SOL is a pair (X, φ) where X is a set of open facilities and φ is an assignment mapping D to $\mathcal{P}(X)$. We call a path in $\varphi(D)$ a *service path.*

For our analysis it would be convenient to represent the total cost of any solution SOL for k-LFLP in the split form $F^{SOL} + C^{SOL}$, where F^{SOL} and C^{SOL} stand for the facility and connection costs, respectively. To break down C^{SOL} further, for any $t = 2, \ldots k$, let C_t^{SOL} denote the total connection cost between open facilities on level $t-1$ and open facilities on level t. Hence $C^{SOL} = \sum_{t=1}^{k} C_t^{SOL}$ where C_1^{SOL} stands for the total connection cost between demand sites and facilities on level 1. Similarly, let F_t^{SOL} denote the total cost to open facilities on level t, and thus $F^{SOL} = \sum_{t=1}^{k} F_t^{SOL}$.

To exploit the cost-split character of the objective function in k-LFLP we modify the standard definition of performance guarantee in the split way:

Definition. A feasible solution SOL of a k-LFLP is called (a, b)-*approximate*

if for any other feasible solution SOL^* of the problem, the cost of SOL is at most $aF^{SOL^*} + bC^{SOL^*}$. An algorithm for a k-LFLP is a (a, b)-*approximation algorithm* if the solution found by the algorithm is (a, b)-approximate.

Our path reduction algorithm was inspired by the observation that the path reduction developed in [2] admits a slight modification implying that any (a, b)-approximation algorithm for UFLP can be translated into a $(a, 3b)$-approximation algorithm for k-LFLP. Therefore, to obtain a good approximation factor for k-LFLP, we have to solve the reduced UFLP in such a way that the performance guarantee pair (a, b) approximately satisfies $a = 3b$. To this point we apply the algorithm of Mahdian et al. [11] to obtain a guarantee pair $(3.27, 1.09)$ for UFLP, which then implies a 3.27-approximation for k-LFLP.

2.2 Parameterized path reduction

We now describe a path reduction with positive parameters a, b that generalizes the reduction in [2] (corresponding to the case $a = b = 1$).

Path reduction with parameters (a, b)**.** Let $\mathcal{M}$ be an instance of k-LFLP. For each $i_1 \in \mathcal{F}_1$ and $t \in \{1, \cdots, |D|\}$, compute a path $p(i_1, t)$ that has the minimum value of $t \cdot bc(p) + af(p)$ over all paths $p \in \mathcal{P}$ starting from i_1. (Note that the problem of finding such paths can be easily reduced to the shortest path problem and there are total $|\mathcal{F}_1| \cdot |D|$ of such paths.) Then, associate with $\mathcal{M}$ an instance $\mathcal{S}$ of UFLP in which the set of demand sites is $\mathcal{D}$, and the set of "facilities" is the set of all pairs (i_1, t) where $i_1 \in \mathcal{F}_1$ and $t \in \{1, \cdots, |D|\}$. In $\mathcal{S}$, for any demand site $j \in D$ and "facility" (i_1, t), the cost of connecting j to (i_1, t) is defined to be $c(j, i_1) + c(p(i_1, t))$; and the cost of opening (i_1, t) is defined to be $f(p(i_1, t))$ (i.e., equal to the cost of opening all facilities on path $p(i_1, t)$). Given a solution $SOLS$ of $\mathcal{S}$, we construct back a solution $SOLM$ of $\mathcal{M}$ as follows: for any $j \in D$, connect j to the service path $p(i_1(j), t)$ such that $(i_1(j), t)$ is the "facility" serving j in $\mathcal{S}$, and open the facilities on all such service paths.

The main result of this subsection is the following theorem.

Theorem 1. *If $SOLS$ is an (a, b)-approximate solution of $\mathcal{I}$, then $SOLM$ is an $(a, 3b)$-approximate solution of $\mathcal{M}$. Furthermore, for any solution SOL*

of $\mathcal{M}$,

$$F^{SOLM} + C^{SOLM} \leq aF^{SOL} + bC_1^{SOL} + 3b\sum_{i=2}^{k} C_i^{SOL}. \tag{2}$$

Therefore, we have

Corollary 1. *Any* (a, b)*-approximation algorithm for solving UFLP yields an* $(a, 3b)$*-approximation algorithm for solving* k*-LFLP.* □

Our proof of the theorem is based on Lemmas 1 and 2 below. The first lemma is nothing but Lemma 2 in [2].

Lemma 1.

$$F^{SOLM} \leq F^{SOLS} \quad \textit{and} \quad C^{SOLM} = C^{SOLS}.$$

□

The second lemma is an improvement of Lemma 4 in [2].

Lemma 2. *For any solution* SOL *of* $\mathcal{M}$*, there exists a corresponding solution* SOL^* *of the reduced* $\mathcal{S}$ *such that*

$$aF^{SOL^*} + bC^{SOL^*} \leq aF^{SOL} + bC_1^{SOL} + 3b\sum_{t=2}^{k} C_t^{SOL}. \tag{3}$$

We first deduce Theorem 1 from the above lemmas.

Proof of Theorem 1. Let SOL^* be any solution of $\mathcal{M}$. By Lemma 2, there exists a corresponding solution SOL of $\mathcal{S}$ such that

$$aF^{SOL} + bC^{SOL} \leq aF^{SOL^*} + bC_1^{SOL^*} + 3b\sum_{t=2}^{k} C_t^{SOL^*} \leq aF^{SOL^*} + 3bC^{SOL^*}.$$

On the other hand, by using Lemma 1 and the fact that $SOLS$ is an (a, b)-approximate solution of $\mathcal{S}$, we have

$$F^{SOLM} + C^{SOLM} \leq F^{SOLS} + C^{SOLS} \leq aF^{SOL} + bC^{SOL},$$

which proves (2). □

To prove Lemma 2 we need the following easy statement, which being a bit stronger than Lemma 3 in [2], has an almost identical proof.

Lemma 3. *Let $\mathcal{I}$ be an instance of k-level FLP and $\overline{SOL}$ be a solution of $\mathcal{I}$. Then $\mathcal{I}$ has a solution $SOL = (X, \varphi)$ such that*

(i) *if in paths $\varphi(j') = (i'_1, \ldots, i'_k)$ and $\varphi(j'') = (i''_1, \ldots, i''_k)$ $i'_l = i''_l$ for some l, then $i'_r = i''_r$ for all $r \geq l$;*

(ii) $C_1^{SOL} = C_1^{\overline{SOL}}$, $\sum_{l=2}^{k} C_l^{SOL} \leq \sum_{l=2}^{k} C_l^{\overline{SOL}}$, $F^{SOL} \leq F^{\overline{SOL}}$. □

The above lemma implies that any solution $\overline{SOL}$ of k-LFLP can be replaced by a solution SOL satisfying (ii) and whose service paths constitute a forest consisting of trees rooted at level k.

Proof of Lemma 2. Let $SOL = (X, \varphi)$ be a solution of $\mathcal{M}$. For any $j \in D$, let $\varphi(j) = \big(i_1(j), \ldots, i_k(j)\big)$. By Lemma 3 we may assume that SOL satisfies property (i) and thus the service paths of SOL constitute a forest consisting of trees rooted at open facilities in $\mathcal{F}_k$.

For every open facility $u \in X_k = X \cap \mathcal{F}_k$ lying on level k, let D_u be the set of demand sites assigned, by φ, to a path finishing in u, and $p(u)$ be a path having minimum value of $c(p)$ among all service paths p ending in u. Also, let $\mu(u)$ be the starting facility of $p(u)$ lying on level 1.

Define a new solution $SOLP = (X, \varphi')$ by reassigning each $j \in D_u$ to the path $p(u)$, i. e., by setting $\varphi'(j) = p(u)$ for all $u \in X_k$. Thus, by definition, $SOLP$ satisfies

$$F^{SOLP} \leq F^{SOL} \tag{4}$$

and

$$C^{SOLP} = \sum_{u \in X_k} \sum_{j \in D_u} \Big(c(j, \mu(u)) + c(p(u)) \Big)$$

By the triangle inequality and the definitions of $p(u)$ and $\mu(u)$,

$$\begin{aligned} c(j, \mu(u)) + c(p(u)) &\leq \big(c(j, i_1(j)) + c(\varphi(j)) + c(p(u)) \big) + c(p(u)) \\ &\leq c(j, i_1(j)) + 3c(\varphi(j)). \end{aligned}$$

Thus we have

$$\begin{aligned} C^{SOLP} \leq & \sum_{u \in X_k} \sum_{j \in D_u} \big(c(j, i_1(j)) + 3c(\varphi(j)) \big) \\ = & C_1^{SOL} + 3 \sum_{t=2}^{k} C_t^{SOL}. \end{aligned} \tag{5}$$

Now, by (4) and (5), it suffices to show that there exists a solution SOL^* of $\mathcal{S}$ such that

$$aF^{SOL^*} + bC^{SOL^*} \leq aF^{SOLP} + bC^{SOLP}. \tag{6}$$

Since the service paths of $SOLP$ are disjoint, we have

$$\begin{aligned} aF^{SOLP} + bC^{SOLP} &= \sum_{u \in X_k} \left(af(p(u)) + b \sum_{j \in D_u} \Big(c(j, \mu(u)) + c(p(u)) \Big) \right) \\ &= \sum_{u \in X_k} \left(af(p(u)) + b|D_u| \cdot c(p(u)) + b \sum_{j \in D_u} c(j, \mu(u)) \right) \\ &= \sum_{u \in X_k} \Big(af(p(u)) + b|D_u| \cdot c(p(u)) \Big) + bC_1^{SOLP}. \end{aligned}$$

Now we define a solution SOL^* of $\mathcal{S}$ by declaring all facilities lying on the paths $p(\mu(u), |D_u|)$, $u \in X_k$, open and by connecting j to the path $p(\mu(u), |D_u|)$ whenever $j \in D_u$. Then we have

$$\begin{aligned} aF^{SOL^*} + bC^{SOL^*} = & \sum_{u \in X_k} \Big(af(p(\mu(u), |D_u|)) + b|D_u| \cdot c(p(\mu(u), |D_u|) \Big) + bC_1^{SOLP} \\ \leq & aF^{SOLP} + bC^{SOLP}. \end{aligned}$$

The last inequality holds because for each $u \in X_k$, by the construction of paths $p(i_1, t)$ in the parameterized path reduction,

$$af(p(\mu(u), |D_u|)) + b|D_u| \cdot c(p(\mu(u), |D_u|) \leq af(p(u)) + b|D_u| \cdot c(p(u)).$$

□

The next subsection analyzes particular values of parameters (a, b) to establish our final result.

2.3 Algorithm Path Reduction&Greedy

To solve the instance $\mathcal{S}$ of UFLP we use the greedy algorithm developed in [11] (in the sequel referred to as Greedy). For completeness, we sketch the algorithm GREEDY below, which essentially is a combination of the algorithms of Jain, Mahdian, and Saberi [9] and Guha and Khuller [7].

Algorithm Greedy:

Phase 1. Given an instance $\mathcal{S}$ of the UFLP, scale up the opening costs of all facilities by a factor of δ (≥ 1) (which is a constant that will be fixed later). Then do the following:

1. At the beginning, all demand sites are *unconnected*, all facilities are *unopened*, and the *budget* of every city j, denoted by B_j, is initialized to 0. At every moment, each demand site j offers some money from its budget to each *unopened* facility i. The amount of this offer is equal to $\max(B_j - c_{ij}, 0)$ if j is unconnected, or $\max(c_{i'j} - c_{ij}, 0)$ if it is already connected to some other facility i'.

2. While there is an unconnected demand site, increase the budget of each *unconnected* demand site at the same rate, until one of the following two events occurs:

   (a) For some unopened facility i, the total offer that it receives from demand sites is equal to the (scaled) cost of opening i. In this case, we open facility i, and connect j to i for every demand site j (connected or unconnected) which has a positive offer to i,

   (b) For some unconnected demand site j, and some facility i that is already open, the budget of j is equal to the connection cost c_{ij}. In this event, we connect j to i.

Phase 2. Scale down the opening costs of facilities back to their original values all at the same rate. If at any point during this process, a facility could be opened without increasing the total cost (i.e., if the opening cost of the facility equals the total amount that the demand sites can save by switching their "service provider" to that facility), then we open the facility and connect each demand site to its closest open facility.

In the following we only need two results from [11].

Lemma 4 ([11]). *Let $\gamma_f^* \geq 1$ and $\gamma_c^* = \sup_k\{z_k\}$, where z_k is the solution of the following optimization program (which we call the factor-revealing LP).*

$$\begin{aligned}
\textit{Maximize} \quad & \frac{\sum_{i=1}^{k} \alpha_i - \gamma_f^* f}{\sum_{i=1}^{k} d_i} \\
\textit{subject to:} & \\
& \alpha_i \leq \alpha_{i+1} \quad \forall\, 1 \leq i < k, \\
& r_{j,i} \geq r_{j,i+1} \quad \forall\, 1 \leq j < i < k, \\
& \alpha_i \leq r_{j,i} + d_i + d_j \quad \forall\, 1 \leq j < i \leq k, \\
& \sum_{j=1}^{i-1} \max(r_{j,i} - d_j, 0) + \sum_{j=i}^{k}, \max(\alpha_i - d_j, 0) \leq f \quad \forall\, 1 \leq i \leq k, \\
& \alpha_j, d_j, f, r_{j,i} \geq 0 \quad \forall\, 1 \leq j \leq i \leq k.
\end{aligned}$$

Then for any $\delta \geq 1$, Algorithm Greedy *is a $(\gamma_f^* + \ln\delta, 1 + \frac{\gamma_c^* - 1}{\delta})$-approximation algorithm for UFLP.*

For any given γ_f^*, one can solve the above linear program to compute γ_c^*. However, since the number of variables here is unbounded, it is unlikely to be computable exactly. In [11], the problem is solved by constructing a feasible solution to the dual of this linear program, which provides an upper bound on γ_c^*.

The crucial result of [11] is the following

Lemma 5 ([11]). *If $\gamma_f^* = 1.11$, then $\gamma_c = 1.78$ is an upper bound on γ_c^*.*

Let $\gamma_f(\delta) = \gamma_f + \ln\delta$ and $\gamma_c(\delta) = 1 + \frac{\gamma_c - 1}{\delta}$ where $\gamma_f = 1.11$, $\gamma_c = 1.78$. By Lemmas 4 and 5, we have the following

Lemma 6. *Algorithm* Greedy *is an $(\gamma_f(\delta), \gamma_c(\delta))$-approximation algorithm for any $\delta \geq 1$.* □

By this lemma, the path reduction algorithm produces a $(\gamma_f(\delta), 3\gamma_c(\delta))$-approximation algorithm for k-LFLP where δ is an arbitrary number ≥ 1. By taking $\delta = 8.67$, one can see that our algorithm, which we will further refer to as Path Reduction&Greedy, finds a solution within a factor of 3.27 of the minimal cost.

Note that the paths $p(i_1, t)$ in the parameterized path reduction can be computed in $O(m^2 n)$ time. On the other hand, the total number of demand sites and facilities in the reduced $\mathcal{S}$ is $n + m_1 n$ and thus GREEDY requires $O(m_1^3 n^3)$ time to solve it. Therefore, the overall running time of PATH REDUCTION&GREEDY is $O(m_1^3 n^3 + m^2 n)$.

We remark that the bound 3.27 cannot be improved much by just using Corollary 1 as a tool box. It is known [7] that for any $x \geq 1$, the existence of $(x, 1 + 2e^{-x})$-approximation algorithm for UFLP would imply $P = NP$. Therefore, the best we could get by using Corollary 1 is 3.236 since $x + 3(1 + 2e^{-x}) \geq 6.472$ for any $x \geq 1$.

3 The recursive path reduction algorithm

A drawback of algorithm PATH REDUCTION&GREEDY is that the approximation factor of 3.27 it provides does not depend on the number of levels k, whereas 1-LFLP admits a 1.52-approximation and the intuition suggests that k-LFLP for small values of k must be much better approximable than the general problem.

In this section, we present an improved combinatorial algorithm for k-LFLP, which we refer to as SPLIT&RECURSION. It is based on a combination of PATH REDUCTION&GREEDY and a recursive reduction of k-LFLP to $(k-1)$-LFLP and UFLP. Algorithm SPLIT&RECURSION runs in time $O(km_1^3 n^3 + km^2 n)$ and achieves an approximation factor $h(k)$, where $h(k) < 3.27$ for any $k \geq 1$ and tends to 3.27 as k tends to ∞. The values of $h(k)$ can be easily computed with an arbitrary accuracy. In particular, $h(2) \approx 2.4211$, $h(3) \approx 2.8446$, $h(4) \approx 3.0565$, $h(5) \approx 3.1678$.

3.1 Definitions and high level description

We first give a few definitions.

For any instance $\mathcal{M}$ of k-LFLP, we define an instance $\mathcal{M}_{k-1}$ of $(k-1)$-LFLP and an instance $\mathcal{S}$ of UFLP (1-LFLP) in the following way:

1. $\mathcal{M}_{k-1}$ is obtained from $\mathcal{M}$ by deleting all facilities on level 1 (or, by opening for free all facilities on level 1). Thus, in $\mathcal{M}_{k-1}$ the set of facilities lying on level r is $\mathcal{F}_{r+1}$, and the connection cost between $j \in D$

and $i_2 \in \mathcal{F}_2$ is

$$\min_{v \in \mathcal{F}_1} \{c(j, v) + c(v, i_2)\}.$$

2. $\mathcal{S}$ is obtained from $\mathcal{M}$ by deleting all facilities on levels greater than 1 (and all edges incident with these facilities), and by doubling all the edge costs between D and $\mathcal{F}_1$.

We are now ready to proceed to a high level description of the algorithm.

In the case $k = 2$, $\mathcal{M}_1$ and $\mathcal{S}$ are both instances of UFLP and we solve them by GREEDY. Note that each $j \in D$ is assigned to a facility $i_2(j) \in \mathcal{F}_2$ by the solution for $\mathcal{M}_1$ and to a facility $i_1(j) \in \mathcal{F}_1$ by the solution for $\mathcal{S}$. On the basis of these solutions we construct a solution for $\mathcal{M}$, denoted by $SOLMS$, by connecting each j to the path $(i_1(j), i_2(j))$.

Note that the straightforward variant of the above construction where the connection costs coincide with the original ones in both instances of UFLP yields a simple factor 3 reduction of 2-LFLP to UFLP. This reduction was first observed by Gimadi [5].

When $k \geq 3$ our algorithm solves $\mathcal{S}$ by applying GREEDY and calls itself to solve $\mathcal{M}_{k-1}$. Now we have that the solution of $\mathcal{S}$ assigns each $j \in D$ to a facility $i_1(j) \in \mathcal{F}_1$ while the solution to $\mathcal{M}_{k-1}$ assigns each $j \in D$ to a path $(i_2(j) \in \mathcal{F}_2, \ldots, i_k(j) \in \mathcal{F}_k)$. In this case the solution $SOLMS$ for $\mathcal{M}$ is constructed by connecting each j to the composite path $(i_1(j), i_2(j), \ldots, i_k(j))$.

However, the constructed solution $SOLMS$ is not yet the output of the algorithm. In addition, we find another solution $SOLPG$ for $\mathcal{M}$ by applying PATH REDUCTION&GREEDY and finally output a solution having lower cost among the two.

By unfolding this recursive description one can easily obtain a conventional implementation as follows. The algorithm applies GREEDY to solve k instances of UFLP obtained from the original instance $\mathcal{M}$ by deleting the facilities on all levels except a fixed one. It then applies PATH REDUCTION&GREEDY to solve $k - 1$ instances of k-LFLP obtained from $\mathcal{M}$ by deleting the facilities on all levels smaller than a fixed one. Finally, in $k - 1$ steps, on the basis of the retrieved solutions, it constructs an output solution.

From the above implementation it is clear that SPLIT&RECURSION can be implemented in $O(k(m_1^3 n^3 + m^2 n))$ time.

3.2 Algorithm Split&Recursion

Now we proceed to a formal description and analysis of the algorithm.

Algorithm Split&Recursion:

Input: An instance $\mathcal{M}$ of k-LFLP.

Output: A solution SOL for $\mathcal{M}$.

`if` $k = 1$ `then`

$SOL :=$ the solution obtained by applying Greedy to $\mathcal{M}$;

`endif`

`if` $k \geq 2$ `then`

Apply Split&Recursion to find a solution $SOLM$ for $\mathcal{M}_{k-1}$ and Greedy to find a solution $SOLS$ for $\mathcal{S}$;

Construct a solution $SOLMS$ for $\mathcal{M}_{k-1}$ by connecting each $j \in D$ to the path $(i_1(j), i_2(j) \ldots, i_k(j))$ whenever j connects to $i_1(j)$ in $SOLS$ and to the path $(i_2(j), \ldots, i_k(j))$ in $SOLM$;

Apply Path Reduction&Greedy to find a solution $SOLPG$ of $\mathcal{M}$;

$SOL :=$ a solution having lower cost among $SOLMS$ and $SOLPG$.

`endif`

The following theorem is the main result of this section.

Theorem 2. *Let $k \geq 2$. For any solution SOL^* of $\mathcal{M}$ and any $\delta \geq 1$, the solution SOL retrieved by* Split&Recursion *satisfies*

$$F^{SOL} + C^{SOL} \leq \gamma_f(\delta) F^{SOL^*} + \theta(k)\gamma_c(\delta) C^{SOL^*} \tag{7}$$

where

$$\theta(k) = 3\left(1 - \frac{1}{2^{k-2}}\right) + \frac{1}{2^{k-3}}.$$

Since $\gamma_f(\delta)$ is a strictly increasing function of δ on the interval $[1, \infty)$ whereas $\theta(k)\gamma_c(\delta)$ is strictly decreasing, the minimum value of

$$\rho_k(\delta) = \max(\gamma_f(\delta), \theta(k)\gamma_c(\delta))$$

is attained at a unique root δ_k of the transcendent equation

$$\gamma_f(\delta) = \theta(k)\gamma_c(\delta).$$

Thus we derive

Corollary 2. SPLIT&RECURSION *is a* $\rho_k(\delta_k)$*-approximation algorithm for* k*-LFLP.*

By using a binary search, it is easy to compute δ_k approximately for every k. This gives

$$\begin{aligned}
\rho_2(\delta_2) &\leq \rho_2(3.71) < 2.4211,\\
\rho_3(\delta_3) &\leq \rho_3(5.66) < 2.8446,\\
\rho_4(\delta_4) &\leq \rho_4(7.0) < 3.0565,\\
\rho_5(\delta_5) &\leq \rho_5(7.66) < 3.1678.
\end{aligned}$$

One can also see that as $k \to \infty$, $\theta(k)$ tends to 3 and the performance factor tends to 3.27 as in algorithm PATH REDUCTION&GREEDY.

Proof of Theorem 2. We proceed by induction on k. Let SOL^* be any solution of $\mathcal{M}$. Then, by Theorem 1,

$$F^{SOLPG} + C^{SOLPG} \leq \gamma_f(\delta)F^{SOL^*} + \gamma_c(\delta)C_1^{SOL^*} + 3\gamma_c(\delta)\sum_{t=2}^{k} C_t^{SOL^*}. \quad (8)$$

Observe that SOL^* induces a solution, $SOLS^*$, to $\mathcal{S}$ and a solution, $SOLM^*$, to $\mathcal{M}_{k-1}$; as SOL^* assigns every demand site j to a facility, say $i_t^*(j) \in \mathcal{F}_t$ for each $t = 1, ..., k$. That is, j in $SOLS^*$ is assigned to $i_1^*(j)$ of $\mathcal{S}$ with connection cost $c(j, i_1^*(j))$, and j in $SOLM^*$ is assigned to $(i_2^*(j), \ldots, i_k^*(j))$ in $\mathcal{M}_{k-1}$ with connection cost at most

$$c(j, i_1^*(j)) + c(i_1^*(j), i_2^*(j)) + c((i_2^*(j), \ldots, i_k^*(j)))$$

from the construction of the connection costs, see (1).

Recall that the connections costs in $\mathcal{S}$ are doubled from the edge costs between D and $\mathcal{F}_1$ in $\mathcal{M}$. Hence, by Lemma 6, we have

$$\begin{aligned}
F^{SOLS} + C^{SOLS} &\leq \gamma_f(\delta)F^{SOLS^*} + 2\gamma_c(\delta)C^{SOLS^*}\\
&= \gamma_f(\delta)F_1^{SOL^*} + 2\gamma_c(\delta)C_1^{SOL^*} \quad (9)
\end{aligned}$$

Assume now that $k = 2$. In this case $\mathcal{M}_{k-1}$ is an instance of UFLP and thus by Lemma 6 and the definition of $\mathcal{M}_{k-1}$,

$$\begin{aligned}
F^{SOLM} + C^{SOLM} &\leq \gamma_f(\delta)F^{SOLM^*} + \gamma_c(\delta)C^{SOLM^*}\\
&\leq \gamma_f(\delta)F_2^{SOL^*} + \gamma_c(\delta)(C_1^{SOL^*} + C_2^{SOL^*}). \quad (10)
\end{aligned}$$

By the construction of $SOLMS$ and the triangle inequality,

$$\begin{aligned} F^{SOLMS} + C^{SOLMS} &= F^{SOLS} + F^{SOLM} + \frac{1}{2}C^{SOLS} + \sum_{j \in D} c(i_1(j), i_2(j)) \\ &\leq F^{SOLS} + F^{SOLM} + \frac{1}{2}C^{SOLS} + \left(\frac{1}{2}C^{SOLS} + C^{SOLM}\right) \\ &= F^{SOLS} + C^{SOLS} + F^{SOLM} + C^{SOLM}, \end{aligned}$$

and thus, by (9) and (10), we have

$$F^{SOLMS} + C^{SOLMS} \leq \gamma_f(\delta) F^{SOL^*} + 3\gamma_c(\delta) C_1^{SOL^*} + \gamma_c(\delta) C_2^{SOL^*}.$$

Since the cost of SOL is at most half as great as the sum of costs of $SOLMS$ and $SOLPG$ (8),

$$F^{SOL} + C^{SOL} \leq \gamma_f(\delta) F^{SOL^*} + 2\gamma_c(\delta) C_1^{SOL^*} + 2\gamma_c(\delta) C_2^{SOL^*},$$

which is nothing but (7) for $k = 2$.

Now, assume that (7) is true for each number of levels smaller than k. By applying to $\mathcal{M}_{k-1}$, which is an instance of $(k-1)$-LFLP, the induction hypothesis we obtain that

$$\begin{aligned} F^{SOLM} + C^{SOLM} \leq & \gamma_f(\delta) \sum_{t=2}^{k} F_t^{SOL^*} + \theta(k-1)\gamma_c(\delta)(C_1^{SOL^*} + C_2^{SOL^*}) + \\ & \theta(k-1)\gamma_c(\delta) \sum_{t=3}^{k} C_t^{SOL^*}. \end{aligned} \tag{11}$$

Again, by the construction of $SOLMS$ and the triangle inequality,

$$F^{SOLMS} + C^{SOLMS} \leq F^{SOLS} + C^{SOLS} + F^{SOLM} + C^{SOLM},$$

and thus, by (9) and (11),

$$\begin{aligned} F^{SOLMS} + C^{SOLMS} \leq & \gamma_f(\delta) F^{SOL^*} + \big(\theta(k-1) + 2\big)\gamma_c(\delta) C_1^{SOL^*} + \\ & \theta(k-1)\gamma_c(\delta) \sum_{t=2}^{k} C_t^{SOL^*}. \end{aligned}$$

Together with (8), this yields

$$F^{SOL} + C^{SOL} \leq \gamma_f(\delta) F^{SOL^*} + \frac{\theta(k-1)+3}{2} \gamma_c(\delta) C^{SOL^*}.$$

Since

$$\theta(k) = \frac{\theta(k-1)+3}{2},$$

(7) follows. □

Finally, we remark that the approximation factors of our algorithms seem to be insensitive to the particular choice of $(\gamma_f, \gamma_c) = (1.11, 1.78)$ used in the above analysis. For example, if one makes use of the pair $(\gamma_f, \gamma_c) = (1, 2)$ (whose correctness was proved for the algorithm presented by Jain et al. [9, 10]), then $h(k)$ is strictly less than 3.301 for any $k \geq 1$ and tends to 3.301 as k tends to ∞. In particular, $h(2) \leq 2.462$, $h(3) \leq 2.882$, $h(4) \leq 3.091$, $h(5) \leq 3.197$.

References

[1] 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 (1999), 161–167.

[2] A. A. Ageev, "Improved approximation algorithms for multilevel facility location problems," *Oper. Res. Letters* 30 (2002), 327–332. The conference version appeared in *Proceedings of 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2002)*, LNCS 2462 (2002), 5–13.

[3] A.F. Bumb and W. Kern, "A simple dual ascent algorithm for the multilevel facility location problem ," *4th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2001)*, LNCS 2129 (2001), 55–62.

[4] N. Edwards, " Approximation algorithms for the multi-level facility location problem," Ph.D. Thesis, School of Operations Research and Industrial Engineering, Cornell University, 2001.

[5] E. Kh. Gimadi, personal communication.

[6] S. Guha, "Approximation algorithms for facility location problems", Ph. D. Thesis, Stanford University, 2000.

[7] S. Guha and S. Kuller, "Greedy strikes back: improved facility location algorithms," *Journal of Algorithms* 31 (1999), 228–248.

[8] S. Guha, A. Meyerson and K. Munagala, "Hierarchical placement and network design problems," *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS 2000)* , 2000, 603–612.

[9] K. Jain, M. Mahdian, and A. Saberi, "A new greedy approach for facility location problems", to appear in Proceedings of the 34th ACM Symposium on Theory of Computing Montreal, Quebec, Canada, May 19-21, 2002.

[10] K. Jain, M. Mahdian, E. Markakis, and A. Saberi, "Greedy Facility Location Algorithms Analyzed using Dual Fitting with Factor-Revealing LP", to appear in Journal of ACM.

[11] M. Mahdian, Y. Ye and J. Zhang, "Improved approximation algorithms for metric facility location problems," *5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2002)*, LNCS 2462 (2002) 229–242.

[12] A. Meyerson, K. Munagala and S. Plotkin, "Cost-distance: two-metric network design," *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS 2000)*, 2000, 624–630.

[13] D. B. Shmoys, "Approximation algorithms for facility location problems," *3rd International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, LNCS 1913 (2000) 27–33.