# Minimizing Movement in Mobile Facility Location Problems

Zachary Friggstad*  Mohammad R. Salavatipour†
Department of Computing Science
University of Alberta
Edmonton, Alberta T6G 2E8, Canada
zacharyf,mreza@cs.ualberta.ca

## Abstract

*In the mobile facility location problem, which is a variant of the classical Uncapacitated Facility Location and $k$-Median problems, each facility and client is assigned to a start location in a metric graph and our goal is to find a destination node for each client and facility such that every client is sent to a node which is the destination of some facility. The quality of a solution can be measured either by the total distance clients and facilities travel or by the maximum distance traveled by any client or facility. As we show in this paper (by an approximation preserving reduction), the problem of minimizing the total movement of facilities and clients generalizes the classical $k$-median problem. The class of movement problems was introduced by Demaine et al. in SODA 2007 [8], where it was observed a simple 2-approximation for the minimum maximum movement mobile facility location while an approximation for the minimum total movement variant and hardness results for both were left as open problems. Our main result here is an 8-approximation algorithm for the minimum total movement mobile facility location problem. We also show that this problem generalizes the classical $k$-median problem using an approximation preserving reduction. For the minimum maximum movement mobile facility location problem, we show that we cannot have a better than a 2-approximation for the problem, unless $P = NP$; so the simple algorithm observed in [8] is essentially best possible.*

## 1 Introduction

Consider the following scenario. There is a company with some manufacturing plants. There are also several retail stores (with different demands) to which the products must be shipped and we are interested in minimizing the cost of shipping. One possibility is to send the products to each retailer from its closest manufacturing plant. Another possibility is to set up a distribution center for each plant (perhaps somewhere else), send the products from that plant to the distribution center (in one shipment) and then for each retailer ship the products from the closest distribution center; this way we save on shipping cost as we might bring the distribution center closer to the set of retailers it is serving and combining the total demands of them into one big shipment to be sent from the plant to the distribution center. This problem can be modeled using the following natural generalizations of the classical $k$-Median and Uncapacitated Facility Location (UFL) problems. Suppose we are given a connected undirected graph $G(V, E)$ with metric distances $d_{ij}$ between every pair of nodes $i, j \in V$. We have a set of clients $C$ with each $c_i \in C$ located at a node. We also have a set of facilities $F$ (corresponding to plants), each located at a node. We want to move each facility and client in the graph to a (possibly different) vertex such that in the final configuration each client is at a node with some facility, while minimizing the total cost of movements of facilities and clients. Formally, we want to assign a destination $v_j$ for each facility $j$ to minimize $\sum_{j \in F} d_{jv_j} + \sum_{i \in C} d_{iv_i}$ where $v_i$ is the nearest facility destination to client $i$. This is called the *minimum total movement mobile facility location* problem, or TM-MFL. If we wish to minimize the maximum distance a client or facility travels then we obtain *minimum maximum movement mobile facility location*, or MM-MFL. Total movement can be thought of as the total amount of resources (e.g. gasoline) consumed by all facilities and clients in reaching a valid solution while maximum movement can be viewed as the time it takes to simultaneously move all units to a valid configuration (e.g. response time). Note that, just like in UFL, we assume that each facility can service any number of clients co-located with it in the final configurations.

These problems fall into a natural class of problems, called movement problems, which were introduced by Demaine et al. [8]. In these types of problems, we are typically

*Supported by NSERC.
†Supported by NSERC and a University Start-up fund

given an instance which contains a weighted graph $G$ together with some pebbles on the vertices (and/or edges) and a desired property $P$; some examples of this property $P$ can be connectivity, independent set, or facility location. We are looking to obtain a movement of pebbles so that the final configuration of pebbles in the graph satisfies the desired property $P$ while minimizing some objective cost function. Some of the natural objective cost functions considered are the total movements of pebbles or the maximum distance a pebble has to move. Many problems of this type arise naturally in other areas, such as operation research, robotics, and design of systems of wireless networks. For instance, suppose each pebble corresponds to a wireless sensor and our goal is to move these sensors around so that they form a connected network. This corresponds to the movement problem with property $P$ being the subgraph induced by the final pebbles' locations being connected. (see e.g. [13, 3] and the references in [8] for more applications).

The movement problems can be defined for different properties $P$. Demaine et al. [8] considered some specific movement problems, including the problem of connectivity (in which our desired property $P$ is that the induced subgraph by the final pebbles' locations is connected), $s,t$-connectivity (in which we want the induced subgraph by the final pebbles' locations contains both $s$ and $t$ in the same connected component), and independent set (pebbles should form an independent set) and gave approximation algorithms and hardness results for each (for different objective functions). They also raised the question of movement mobile facility location problem. For the minimum maximum movement mobile facility location (MM-MFL), they [8] observed that there is a simple 2-approximation and asked whether this can be improved. They also left the problem of finding a good approximation algorithm for the minimum total movement mobile facility location (TM-MFL) as an open question. In this paper, we answer both these questions. For MM-MFL, we show that it is NP-hard to obtain better than a 2-approximation. The main contribution of this paper is to present a constant factor approximation algorithm for the TM-MFL defined earlier. As we will see, this problem in fact generalizes the classical $k$-median problem. We show that there is an approximation preserving reduction from $k$-median to minimum total movement facility location.

**Related Works:** In the classical (uncapacitated) facility location problem UFL, we are given a graph $G(V, E)$ with metric costs $d_{ij}$ on the edges, a set of clients $C \subseteq V$, and a set of facilities $F \subseteq V$ with each $i \in F$ having an opening cost $f_i$. The goal is to open some of the facilities and assign each client to an open facility such that the total cost of opening facilities plus the costs of clients traveling to open facilities is minimized. The first approximation algorithm for facility location had ratio $O(\log n)$ and is due to Hochbaum [12]. Shmoys, Tardos, Aardal [21] were the first to give a constant ratio approximation for this problem; their algorithm had ratio 3.16. Later, in a series of papers several constant approximation algorithm were obtained for this problem with better ratios (see [11, 7, 14, 16, 1, 22, 19, 4] and references in [4]). The best known algorithm has ratio 1.5 [4]. Guha and Khuller [11] showed that, unless $NP \subseteq \mathrm{DTIME}(n^{\mathrm{polylog}(n)})$, there is no better than a 1.463-approximation for UFL. Several variations of the facility location problem have been studied such as capacitated facility location, in which there is a capacity on the number of clients that can be served at each facility $f_i$ (e.g. see [18] and the references there).

Another well-studied related problem is the classical $k$-median. In the $k$-median problem there is no opening cost for facilities but we can open up to $k$ facilities. In the more general setting of $k$-median, each client $c_i \in C$ can have a (positive) demand $D_i$ and the cost of serving this demand at location $j$ (if there is facility there) is $D_i \cdot d_{ij}$. The best known approximation algorithm for $k$-median uses local search heuristics and has ration $3 + \epsilon$, for any constant $\epsilon > 0$, due to Arya et al [1]. On the other hand, Jain, Mahdian, and Saberi [15] showed that unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{O(\log \log n)})$, then there is no 1.735-approximation algorithm for the $k$-median problem.

A common generalization of both UFL and $k$-median is a problem called $k$-facility location. The input to this problem is like $k$-median, except that each facility has an openning cost. The goal is to open at most $k$ facilities and assign clients to them to minimize the total cost. Devanur et al. [9] have shown that $k$-facility location has a locality gap of at most 5.

Demaine et al. [8] considered some classes of movement problems. For the property of forming a connected induced subgraph, they obtained approximation algorithms with ratios $O(\sqrt{m/OPT})$ and $\tilde{O}(\min\{n, m\})$ for minimum maximum movement and total movement, respectively (where $n = |V(G)|$ and $m = |E(G)|$), and hardness of $\Omega(n^{1-\epsilon})$ for the total movement (they use the term "sum" to refer to what we call "total" movement). They also considered variations in which pebbles need to establish connectivity between two given nodes $s, t$, or to form an independent set (on the plane $\mathbb{R}^2$).

**Our results:** We consider both the TM-MFL and MM-MFL problems. We can assume each client represents one unit of demand (to be serviced). If there are $D_i$ clients at a vertex $v$, without loss of generality, we can assume that all these clients have the same final destination in an optimal solution. Therefore, we can combine them all into a single client $c_i$ whose demand is equal to $D_i \geq 1$ and the cost of moving client $c_i$ a distance of $\ell$ is $D_i \cdot \ell$. For this reason, we consider the slightly more general version of TM-MFL in which each client $c_i \in C$ has a demand $D_i \geq 1$ and the goal is to assign a destination $v_j$ for each facility $j$ to minimize $\sum_{j\in F} d_{jv_j} + \sum_{i\in C} D_i d_{iv_i}$ where $v_i$ is the nearest facility destination to client $i$. As mentioned earlier, the model we consider here is the same as the one defined by Demain et al. [8], which is an extension of Uncapacitated Facility Location (UFL) and $k$-Median, in the sense that each facility can serve an unbounded number of clients, and the cost associcated with it is independent of the number of demands (clients) it serves (in UFL, the cost is the openning cost $f_i$, here it is the cost of moving it to its final destination). Note that the demand (number of individual clients) per node is irrelevant in MM-MFL since we are only concerned with the maximum distance. For TM-MFL restricted to trees it is possible to obtain a pseudo-polynomial time exact algorithm where the demands are polynomial in the size of the input. Using the fact that every graph metric can be probabilistically embedded into a tree with distortion $O(\log n)$ [2, 10], this yields an $O(\log n)$-approximation for TM-MFL in general graphs in pseudo-polynomial time. However, obtaining a true $O(\log n)$-approximation seems non-trivial. For example, unlike the classical facility location problem, the natural greedy algorithm that tries to find good partial solutions iteratively fails. Our main result in this paper is the following.

**Theorem 1.1.** *There is a polynomial time deterministic 8-approximation algorithm for minimum total movement mobile facility location (TM-MFL).*

This algorithm is based on rounding an optimal solution to an IP/LP relaxation of the problem in five major rounds. Each round brings the solution closer to an integer solution. This algorithm is inspired by the work of Charikar et al. [6] but uses several new ideas, such as the total unimodularity of the matching polytope as well as an augmenting path argument to obtain a half-integer solution. Although the algorithm is fairly involved, we believe some of the ideas developed here might be useful in solving other combinatorial optimization problems. We can also show that any natural local search algorithm that performs a bounded number of exchange or switch operations at each iteration will have an unbounded ratio (see the remark after Theorem 1.4). This theorem is complemented by the following whose proof follows almost immediately from the proof of APX-hardness for uncapacitated facility location by Guha and Khuller [11]:

**Theorem 1.2.** *The minimum total movement mobile facility location (TM-MFL) problem is APX-hard.*

We also present an approximation preserving reduction from $k$-median to TM-MFL. Note that the best approximation algorithm for $k$-median has ratio $3 + \epsilon$ [1].

**Theorem 1.3.** *If there is an $\alpha$-approximation for TM-MFL then there is an $(\alpha + o(1))$-approximation for the $k$-median problem.*

Jain, Mahdian, and Saberi [15] proved that there is no 1.735-approximation algorithm for $k$-median unless $NP \subseteq \mathrm{DTIME}(n^{\mathrm{polylog}(n)})$. By Theorem 1.3, the same hardness holds for TM-MFL.

For the MM-MFL problem, there is a simple 2-approximation algorithm (as observed in [8]) as follows: do not move the facilities; only move each client to the nearest facility. It is easily seen that the maximum distance traveled in this solution is at most twice the optimum. We show that this is essentially best possible:

**Theorem 1.4.** *For any $\epsilon > 0$, there is no $(2 - \epsilon)$-approximation algorithm for the minimum maximum movement mobile facility location problem (MM-MFL) unless $P = NP$.*

**Remark:** Since the best known approximation algorithm for $k$-median uses local search [1] and given the similarity of the TM-MFL problem to $k$-median (e.g. by Theorem 1.3), it is natural to guess that local search technique might also be useful to design an approximation algorithm for TM-MFL. However, we can construct examples that show that a class of natural local search algorithms that perform a bounded number of exchange or switch operations at each iteration will have an unbounded ratio; note that this class includes the local search constant approximation algorithm of [1] for $k$-median. More specifically, we can show the following. First observe that if we fix the destinations of the set of facilities then the solution for clients is obvious (each client must go to the nearest vertex which has a facility). Now consider any algorithm that, for a bounded value $p$, iteratively performs one or both of the following operations as long as it improves the quality of the solution:

1. select a subset $k \le p$ of facilities $f_{i_1}, \ldots, f_{i_k}$ and a subset of size $k$ of destinations for them $v_1, \ldots, v_k$, respectively; move $f_{i_j}$ to $v_j$, for each $1 \le j \le k$.
2. select a subset $k \le p$ of facilities $f_{i_1}, \ldots, f_{i_k}$ and let $v_1, \ldots, v_k$ be their current location, then find a permutation $\pi : [k] \to [k]$ and move each $f_{i_j}$ to $v_{\pi(j)}$, for all $1 \le j \le k$.

Then there are examples for which the algorithm above will have unbounded approximation ratio, we will give such an example in Section 4.

The rest of the paper is organized as follows. For ease of exposition and lack of space, we present a proof for a slightly weaker version of Theorem 1.1 in the next section and postpone the proof of the main theorem to full version of the paper. The proofs of Theorems 1.3 and 1.4 appear in Section 3.

# 2 A Randomized 16-Approximation Algorithm

In this section, we present a randomized 16-approximation algorithm for the minimum total movement mobile facility location. This algorithm uses randomized rounding of the optimal fractional solution obtained from a solving a natural IP/LP formulation. It can be easily de-randomized using the method of conditional probablities.

Recall that in TM-MFL, we have a graph $G(V, E)$ with metric costs $d_{ij}$ on the edges, a set of clients $C$ each having a demand $D_i \geq 1$, and a set of facilities $F$. Note that, since we do not need more than one facility on any node in the final configuration, we assume that each node has at most $|V|$ facilities so $|F| \leq |V|^2$. Furthermore, we then assume (after the previous observation) that each facility is located on a node with no other facilities or clients; this can be enforced by transforming the general problem of facilities sharing a node by creating a dummy-node for each facility and connecting it to the original node with cost 0. Thus we can assume $F \subseteq V$. Also, we can assume $C \subseteq V$ by combining the demands of clients in any node into one client since clients starting on the same node can be moved to the same destination in the optimal solution.

## 2.1 Outline of the Algorithm

Our starting point is an integer programming formulation of the problem. Define indicator variables $x_{iv}$ for each $i \in C$ and $v \in V$, and $y_{jv}$ for each $j \in F$ and $v \in V$; variables $x_{iv}$ and $y_{jv}$ will be 1 if client $i$ or facility $j$ is moved to location $v$, respectively, and 0 otherwise. Then the goal is to optimize to following program:

$$\begin{array}{llll} \text{minimize :} & \sum_{i\in C}\sum_{v\in V} x_{iv}D_i d_{iv} & + & \sum_{j\in F}\sum_{v\in V} y_{jv}d_{jv} \\ \text{such that :} & \sum_{v\in V} x_{iv} & = 1 & \forall i \in C \\ & \sum_{v\in V} y_{jv} & = 1 & \forall j \in F \\ & \sum_{j\in F} y_{jv} & \geq x_{iv} & \forall i \in C, v \in V \\ & x_{iv} \in & \{0,1\} & \forall i \in C, v \in V \\ & y_{jv} \in & \{0,1\} & \forall j \in F, v \in V \end{array}$$

The first two constraints ensure that a client or facility has a unique destination vertex while the third ensures that any vertex that is the destination of some client is also the destination of some facility. We say a location $v$ is *covered* if there is at least one facility assigned to it (so we can move any client to be served at $v$). We obtain a linear program (LP) relaxation of this problem by relaxing the last two constraints to non-negativity constraints $x_{iv} \geq 0$ and $y_{jv} \geq 0$. Since the size of this LP is polynomial in the size of the input we can compute the optimum fractional solution $(\overline{x}, \overline{y})$ with objective function value $OPT_f$. For each client $i$ define $\overline{C}_i = \sum_v x_{iv} d_{iv}$ and for each facility $j$ define $\overline{F}_j = \sum_v y_{jv} d_{jv}$. Note that $\overline{C}_i$ and $\overline{F}_j$ are the total costs of moving a unit of demand of client $i$ and facility $j$, respectively. Denote $\sum_i \overline{C}_i D_i$ as $\overline{C}$ and $\sum_j \overline{F}_j$ as $\overline{F}$; so $\overline{C} + \overline{F} = OPT_f$. Our randomized algorithm produces an integral solution of (expected) cost at most $16\overline{C} + 4\overline{F} \leq 16 \cdot OPT_f$.

The algorithm has five phases, starting from the optimal fractional solution $(\overline{x}, \overline{y})$, where each phase brings the current solution closer to an integer solution while keeping a bound on the cost increases. We begin with a summary of each phase. Since the values of $(\overline{x}, \overline{y})$ change frequently throughout the algorithm, we adopt the following notation. For each step $p$, $(\overline{x}^{(p)}, \overline{y}^{(p)})$ will denote the assignments of clients and facilities to locations after step $p$. Similarly, we will let $\overline{C}_i^{(p)}$ and $\overline{F}_j^{(p)}$ denote the respective costs of moving one unit of demand of client $i$ and moving facility $j$ under the assignment $(\overline{x}^{(p)}, \overline{y}^{(p)})$. Finally, we let $\overline{C}^{(p)} = \sum_i \overline{C}_i^{(p)} D_i$ and $\overline{F}^{(p)} = \sum_j \overline{F}_j^{(p)}$.

**Step 1: Clustering of clients:** To start, we will create a modified instance of the original problem by moving demands between clients and removing clients with zero demand so that different locations with non-zero client demands are far apart. More specifically, for all pairs of clients $i \neq i'$ we want to make sure that $d_{ii'} > 4 \cdot \max\{\overline{C}_i, \overline{C}_{i'}\}$. This will be guided by the values of $(\overline{x}, \overline{y})$ so that the cost of the new instance under the assignments of $(\overline{x}, \overline{y})$ is at most $OPT_f$ and so we can recover an integer solution to the original problem from an integer solution to the modified problem by paying only a constant factor of $\overline{C}$.

**Step 2: Relocation of facilities:** The next step is to ensure that each location $v$ with $x_{iv}^{(2)} > 0$ for some $i \in C$ is the initial location of some client $i'$. That is, at the end of Step 2, $x_{iv}^{(2)} > 0$ implies there is some $i'$ where client $i'$ starts at node $v$. Based on the previous step of clustering the demands and on how we perform this step, we will now be able to say that $x_{ii}^{(2)} \geq \frac{1}{2}$, so less than half of each client $i$ must be served at a location $v$ different from $i$ and that this location is the location of another client. Moreover, each client has this amount served at the nearest $i'$ to $i$ while breaking ties by choosing the client $i'$ with lowest index. Let this closest client to $i$ be denoted by $\phi(i)$. Finally, we remove useless facilities so that each location $v$ now has $\sum_j y_{jv}^{(2)} \leq 1$.

**Step 3: Getting a half-integer solution:** The third step is more involved. Here we obtain a half-integer solution through two sub-processes. The first ensures that for each location $v$, $\sum_j y_{jv}^{(3)} = \frac{a}{2}$ for some integer $a$ by redirecting facility assignments using an augmenting path. The second relies on a matching polytope argument and uses a minimum weighted perfect matching algorithm (in a bipartite graph) to ensure that we can assume each individual $y_{jv}^{(3)} \in \left\{0, \frac{1}{2}, 1\right\}$. At this stage, we will have half-integer values for all of $x_i^{(3)}$ and $y_j^{(3)}$ variables.

**Step 4: Modifying the half-integer solution:** The final step of the algorithm has each facility uniformly randomly rounded to one of its at most 2 (fractional) destinations. However, there are some events that we will consider as *"bad"*. The purpose of step 4 is to reassign some facility variables in a way that removes the possiblity of these bad events while maintaining their half-integrality and incurring only a constant-factor blowup in their cost.

**Step 5: Randomized Rounding** As described before, we randomly round each facility location to one of its at most 2 (fractional) destination locations with equal probability. Then we move each client $i$ to $\phi(i)$ and from there to $\phi(\phi(i))$ if there is no facility assigned there, and so on, until the client reaches a covered location.

## 2.2 Clustering of Clients

This phase is similar to the first step in [6]. Recall that we start from an optimum LP solution and $\overline{C}_i = \sum_v x_{iv} d_{iv}$. We modify the instance in such a way that the current fractional solution is also a feasible solution for the new instance, and given an integer solution for the new instance, we can obtain an integer solution for the original instance with a bounded increase in the cost. Without loss of generality, assume that $\overline{C}_1 \leq \overline{C}_2 \leq \cdots \leq \overline{C}_{|C|}$. We assign $(\overline{x}^{(1)}, \overline{y}^{(1)}) \leftarrow (\overline{x}, \overline{y})$ and cluster the demands of clients by the following procedure:

**for** $i = 1 \ldots |C|$ **do**
  **if** $\exists\, i' < i$ such that $D_{i'} > 0$ and $d_{ii'} \leq 4\overline{C}_i$ **then**
    $D_{i'} \leftarrow D_{i'} + D_i$
    $D_i \leftarrow 0$ ($i$ is no longer a client)

In the following three lemmas we show how to find a good integer solution of the original instance from a solution of this new instance. The first lemma describes how to obtain an integer solution to the original instance from an integer solution to the new instance with an increase of $4\overline{C}$ in cost. The second lemma expresses the fact that this new problem does not get worse in its objective function value under the assignments $(\overline{x}, \overline{y})$. The final lemma states that any two clients are far from each other; a property that is used in obtaining an integer solution of the new instance. The proofs of the first two lemmas are fairly simple and are omitted, while the third is immediate from the clustering procedure.

**Lemma 2.1.** *Any integer solution of cost $T$ to the modified instance can be turned into an integer solution of the original problem with cost at most $T + 4\overline{C}$.*

**Lemma 2.2.** $\overline{C}^{(1)} \leq \overline{C}$ *and* $\overline{F}^{(1)} = \overline{F}$.

**Lemma 2.3.** *Any two clients $i$ and $i'$ in the modified instance have $d_{ii'} > 4 \cdot \max\left\{\overline{C}_i^{(1)}, \overline{C}_{i'}^{(1)}\right\}$.*

From now on, we will be dealing with the modified instance of the problem.

## 2.3 Relocation of Facilities

First, initialize $(\overline{x}^{(2)}, \overline{y}^{(2)}) \leftarrow (\overline{x}^{(1)}, \overline{y}^{(1)})$. Now, consider each location $v$ with $x_{iv}^{(2)} > 0$ for some client $i$ (i.e. there is some client being served fractionally at $v$) but there is no client demand located at $v$. Let $i' \in C$ be any closest client to this location, i.e. $d_{i'v} \leq d_{iv}$ for all other clients $i \in C$. We are going to relocate the clients being served at $v$ to location $i'$ and take the facilities that cover $v$ with them. Let $M = \frac{\max_i x_{iv}^{(2)}}{\sum_j y_{jv}^{(2)}}$ be the fraction of the facility coverage at $v$ that is required to cover all clients $i$ being served at $v$ (it may be that $\sum_j y_{jv}^{(2)} > x_{iv}^{(2)}$ for all $i \in C$, for example when $d_{jv} = 0$ for many facilities $j$ but we will remove all such occurrences after this step). For all clients $i$ and facilities $j$, assign: $x_{ii'}^{(2)} \leftarrow x_{ii'}^{(2)} + x_{iv}^{(2)}$, $x_{iv}^{(2)} \leftarrow 0$, $y_{ji'}^{(2)} \leftarrow y_{ji'}^{(2)} + M \cdot y_{jv}^{(2)}$, and $y_{jv}^{(2)} \leftarrow (1 - M) \cdot y_{jv}^{(2)}$.

**Lemma 2.4.** $\overline{C}^{(2)} \leq 2\overline{C}^{(1)}$ *and* $\overline{F}^{(2)} \leq \overline{F}^{(1)} + \overline{C}^{(1)}$.

*Proof.* Consider a vertex $v$ as above where the assignments at $v$ were moved to a nearest client $i'$. For each client $i$ the cost increases by at most $D_i \cdot x_{iv}^{(1)} \cdot d_{vi'} \leq D_i \cdot x_{iv}^{(1)} \cdot d_{vi}$, where we use the fact that $d_{vi'} \leq d_{vi}$ (by definition of $i'$). This fraction of the client's assignment will not be moved again since it is moved to a client location. Therefore, summing over all $i$ and $v$, it follows that the cost increase for clients is at most $\overline{C}^{(1)}$.

For each vertex $v$ we know there is a client $i$ with $x_{iv}^{(1)} = M \cdot \sum_j y_{jv}^{(1)}$. The cost increase incurred by moving facility assignments to $v$ is:

$$\begin{aligned} M \cdot \sum_{j \in F} y_{jv}^{(1)} (d_{ji'} - d_{jv}) \quad &\leq \quad M \cdot \sum_{j \in F} y_{jv}^{(1)} d_{i'v} \\ = d_{i'v} \cdot M \cdot \sum_{j \in F} y_{jv}^{(1)} \quad &= \quad x_{iv}^{(1)} \cdot d_{i'v} \\ \leq x_{iv}^{(1)} \cdot d_{iv} \quad &\leq \quad x_{iv}^{(1)} \cdot d_{iv} \cdot D_i, \end{aligned}$$

where the first inequality uses the fact that $d_{ji'} \leq d_{jv} + d_{vi'}$ by triangle inequality, and the last one uses the assumption that $D_i \geq 1$. Notice that when an assignment of a facility is moved from some $v$ to some $i'$ then that fraction of the assignment will never move again in this step. Furthermore, each $x_{iv}^{(1)}$ fraction of a client will be used at most once in bounding the facility cost increase. Therefore, summing this change in cost over all $v$ shows the cost increase for both clients and facilities is bounded by $\overline{C}^{(1)}$. $\square$

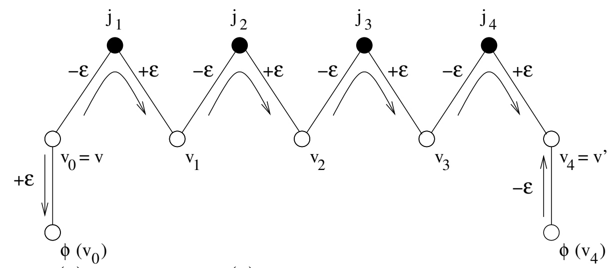
Even more can be said about the structure of the current solution. Using a simple averaging argument we can prove:

**Lemma 2.5.** *For all clients $i$: $x_{ii}^{(2)} \geq \frac{1}{2}$. In other words, each client has less than half of its assignment being served at a different location than its own.*

Next, for each vertex $v$ with $D_v = 0$ and each facility $j$, if $y_{jv}^{(2)} > 0$ then move this amount back to the location of $j$ at no additional cost. This can happen if at the beginning of this phase, $M < 1$; therefore after relocating the facility values assigned to vertex $v$ we still have $\sum_j y_{jv}^{(2)} > 0$. Then, for each client $i$ if $\sum_j y_{ji}^{(2)} > 1$ we can move coverage from facilities from $y_{ji}^{(2)}$ to $y_{jj}^{(2)}$ until $\sum_j y_{ji}^{(2)} = 1$ at no extra cost. Since we assume that each facility starts on a location with no other facilities or clients then this is always possible. Thus, in the current solution we have that the only vertices with non-zero coverage are those that are either a client or facility location, $\sum_j y_{ji}^{(2)} \leq 1$ for all locations $v$ and, for all clients $i$, $x_{ii}^{(2)} \geq \frac{1}{2}$ and $\sum_{i' \in C} x_{ii'}^{(2)} = 1$. We can assume that the remaining $1 - x_{ii}^{(2)} \leq \frac{1}{2}$ fraction of each client $i$ not being served at its own location is being served at the nearest client. Denote this client as $\phi(i)$ while breaking ties by the lowest index. Also assume that each client uses the coverage at its own location to the maximum amount. That is, for each $i$, we can assume that $x_{ii}^{(2)} = \sum_j y_{ji}^{(2)}$ by moving $\sum_j y_{ji}^{(2)} - x_{ii}^{(2)}$ from $x_{i\phi(i)}^{(2)}$ to $x_{ii}^{(2)}$ at no additional cost.

## 2.4 Getting a Half-Integer Solution

In this phase our goal is to ensure that the value of each $x_{iv}^{(3)}$ and $y_{jv}^{(3)}$ is in $\{0, \frac{1}{2}, 1\}$. Start with $(\overline{x}^{(3)}, \overline{y}^{(3)}) \leftarrow (\overline{x}^{(2)}, \overline{y}^{(2)})$. We say that a location $v$ is covered half-integrally if $\sum_j y_{jv}^{(3)} \in \{0, \frac{1}{2}, 1\}$. Given the assignment $(\overline{x}^{(3)}, \overline{y}^{(3)})$, we construct a weighted bipartite graph $B$ in the following manner. For each location $v$ create a vertex on one side of the bipartition and for each facility $j$ create a vertex on the other side of the bipartition. We connect $v$ to $j$ in $B$ with weight $y_{jv}^{(3)}$ only if this weight is positive. The edge weights in any connected component of $B$ must sum to an integer since the sum of the weights of the edges incident to any particular facility must be 1. Therefore, if there is a location $v$ that is not covered half-integrally in the current assignment $(\overline{x}^{(3)}, \overline{y}^{(3)})$, then there must be another location $v'$ in the same connected component as $v$ that is not covered half-integrally in $B$.




$$
\begin{array}{lcll}
y_{j_m v_{m-1}}^{(3)} & \leftarrow & y_{j_m v_{m-1}}^{(3)} - \epsilon & \forall\, 1 \leq m \leq k \\
y_{j_m v_m}^{(3)} & \leftarrow & y_{j_m v_m}^{(3)} + \epsilon & \forall\, 1 \leq m \leq k \\
x_{ii}^{(3)} & \leftarrow & x_{ii}^{(3)} - \epsilon & \text{if } v \text{ is a client } i \\
x_{i\phi(i)}^{(3)} & \leftarrow & x_{i\phi(i)}^{(3)} + \epsilon & \text{if } v \text{ is a client } i \\
x_{i'i'}^{(3)} & \leftarrow & x_{i'i'}^{(3)} + \epsilon & \text{if } v' \text{ is a client } i' \\
x_{i'\phi(i')}^{(3)} & \leftarrow & x_{i'\phi(i')}^{(3)} - \epsilon & \text{if } v' \text{ is a client } i'
\end{array}
$$

**Figure 1. Changing fractional values over a path between two clients with non-half-integer coverage.**

While there is still a location $v$ that is not covered half-integrally we execute the following procedure. Find a path $v = v_0, j_1, v_1, j_2, v_2, \ldots, v_{k-1}, j_k, v_k = v'$ in the bipartite graph $B$ constructed from $v$ (using current $(\overline{x}^{(3)}, \overline{y}^{(3)})$) to some other location $v'$ that is not covered half-integrally. Define $\alpha_0 = d_{i\phi(i)} D_i$ if $v$ is some client $i$ and otherwise say $\alpha_0 = 0$. Similarly define $\alpha_k$ for $v'$. These $\alpha_0, \alpha_k$ quantities express the cost of serving one unit of demand for the clients at locations corresponding to $v$ and $v'$ if there is no facility coverage at their location.

Let $\tau$ be a constant which we will specify after the entire algorithm is presented. Since we could consider this path in the reverse order, without loss of generality, assume that:

$$\alpha_0 + \tau \sum_{m=1}^{k} d_{j_m v_m} \leq \alpha_k + \tau \sum_{m=1}^{k} d_{j_m v_{m-1}} \qquad (1)$$

What we plan to do is shift some coverage from $v$ to $v'$ through this path, by simultaneously increasing each $y_{j_m v_m}^{(3)}$ and decreasing each $y_{j_m v_{m-1}}^{(3)}$ (at the same rate) until one of the edges in the path has value 0 (*i.e.* the edge disappears from $B$) or one of the endpoints $v$ or $v'$ is covered half-integrally. Let $\epsilon = \min\{y_{j_m, v_{m-1}}^{(3)} | 1 \leq m \leq k\}$. If $v$ is a client $i$, then update $\epsilon \leftarrow \min\{\epsilon, x_{ii}^{(3)} - \frac{1}{2}\}$. Finally, update $\epsilon \leftarrow \min\{\epsilon, 1 - \sum_j y_{jv'}^{(3)}\}$ (this is $1 - x_{i'i'}^{(3)}$ if $v'$ is a client $i'$). Now, by our construction of $B$ and the assumption that neither $v$ nor $v'$ are covered half-integrally, we have $\epsilon > 0$. We perform updates to $(\overline{x}^{(3)}, \overline{y}^{(3)})$ as in Figure 1.

Since a new edge in $B$ can never be introduced by this method and all half-integrally covered locations remain so, then there is a polynomial upper-bound on the number of times we must perform this re-assignment of facilities. Denote the difference $\epsilon(\sum_{m=1}^{k}(d_{j_m v_m} - d_{j_m v_{m-1}}))$ by $\delta$. After a step is performed, the total cost of all clients $\overline{C}^{(3)}$ is changed by $\epsilon(\alpha_0 - \alpha_k)$ which is at most $-\tau\delta$ by (1). Similarly, the change in the cost of the facilities is $\delta$. Letting $\Delta$ be the sum of the $\delta$ values over all executions of such an update, we see that $\overline{C}^{(3)} \le \overline{C}^{(2)} - \tau\Delta$ and $\overline{F}^{(3)} = \overline{F}^{(2)} + \Delta$.

The previous process obtains a half-integer solution in that each location $v$ has $\sum_j y_{jv}^{(3)} \in \{0, \frac{1}{2}, 1\}$. However, it is not necessarily true that each $y_{jv}^{(3)}$ is a half-integer. We rectify this situation with a matching.

**Lemma 2.6.** *If there is an assignment $(\overline{x}, \overline{y})$ such $\sum_j y_{jv}$ is a half-integer for all locations $v$, then we can find an assignment $(\overline{x}', \overline{y}')$ with all $y'_{jv}$ being half-integer where neither the client nor the facility costs increase.*

*Proof.* We split all locations $v$ with $\sum_j y_{jv} = \frac{a(v)}{2}$ into locations $v_1, \ldots, v_{a(v)}$ (note that $a(v) \in \{0, 1, 2\}$). Split each facility $j$ into nodes $j_1$ and $j_2$. Finally, for each original location $v$ with $a(v) > 0$ and each original facility $j$, set $y_{j_1 v_d} = y_{j_2 v_d} = \frac{y_{jv}}{a(v)}$ for $1 \le d \le a(v)$ and keep the original distances. Notice that for each new facility, the sum of fractional values over the edges incident to it is still 1 and each new location is fractionally covered with value exactly 1. We now have a fractional matching between the $2|F|$ new facilities and the $2|F|$ new locations of cost $2\overline{F}$ (since we doubled the edges). By total-unimodularity of the matching polytope, we can assume that there is an integral matching of at most the same cost. We can find such a matching using a minimum weight perfect matching in bipartite graphs algorithm (e.g. [20]). Consider the new assignment values defined by this matching; all $y_{j_\alpha v_d}$'s are either 0 or 1.

Now in the original problem (before splitting) let $y'_{jv} \leftarrow \frac{1}{2}\sum_{d=1}^{a(v)}(y_{j_1 v_d} + y_{j_2 v_d})$ for each location $v$ and facility $j$. Since we had an integer matching with the new locations and facilities, we now have that $y'_{jv} \in \{0, \frac{1}{2}, 1\}$. Each new location $v_d$, for all $1 \le d \le v_{a(v)}$, was covered with weight 1 in the integer matching so each original location $v$ still has $\sum_j y'_{jv} = \frac{a(v)}{2}$. The weights of the assignments in the integer matching were halved to restore the original problem, so now the current cost of the facilities is at most $\overline{F}$. The client assignments $\overline{x}$ do not change so the client costs do not increase. □

Applying Lemma 2.6 to $(\overline{x}^{(3)}, \overline{y}^{(3)})$, we obtain an assignment to $x, y$ variables in which each $y_{jv}^{(3)}$ is half-integer, for all $j, v$. Since each client $i$ has $x_{ii}^{(3)} \ge \frac{1}{2}$ and uses its own coverage to the maximum amount, it follows that each $x_{ii}^{(3)} \in \{\frac{1}{2}, 1\}$ and if $x_{ii}^{(3)} = \frac{1}{2}$ then $x_{i\phi(i)}^{(3)} = \frac{1}{2}$.

## 2.5 Modifying the Half-Integer Solution

We construct an auxiliary directed graph $H$ which has a vertex $v_i$ for each client $i$ and a directed edge from $v_i$ to $v_{\phi(i)}$ with weight $d_{i\phi(i)}$. So each vertex in $H$ has outdegree exactly one and by the definition of $\phi$, the edge weights are non-increasing in any walk on $H$, which means any directed cycle of $H$ must have all edge weights being the same. Moreover, since $\phi(i)$ was defined by breaking ties with lower-indexed clients then all cycles in $H$ have length 2. So, $H$ can be viewed as a collection of connected components each of which is a unicyclic graph consisting of a directed tree with a 2-cycle at the root and all other edges being oriented toward the root.

We define two types of *bad* clients. Call each client $i$ a *type 1* bad client if $x_{ii}^{(3)} = 1$ and for two distinct facilities $j$ and $j'$, $y_{ji}^{(3)} = y_{j'i}^{(3)} = \frac{1}{2}$. Also, a pair of clients $i$ and $i'$ are called *type 2* if $\phi(i) = i'$, $\phi(i') = i$ and $x_{ii}^{(3)} = x_{i'i'}^{(3)} = \frac{1}{2}$ but two different facilities $j$ and $j'$ are such that $y_{ji}^{(3)} = y_{j'i'}^{(3)} = \frac{1}{2}$. The remaining locations are *good* ones. Recall that the final step of the algorithm will round each facility $j$ to location $v$ with probability $y_{jv}^{(4)}$. With this in consideration, we see that type 1 clients are those where $\overline{C}_i^{(3)} = 0$ but $i$ might not receive a facility after the randomized rounding of facilities (and so has to be served at the nearest location with a facility), thus incurring a positive cost increase. We make a modification to the half-integer solution such that this does not happen in the next step. We will bound the expected cost of each client by considering the expected distance a client $i$ has to move along the sequence of locations $\phi^{(0)}(i), \phi^{(1)}(i), \phi^{(2)}(i) \ldots$ until it reaches a location that received a facility. Here we define $\phi^{(k)}(i)$ recursively as $\phi^{(0)}(i) = i$ and $\phi^{(k+1)}(i) = \phi(\phi^{(k)}(i))$. The problem with type 2 clients is that they will never reach a location with a facility if both of their half-covering facilities are assigned elsewhere in the randomized rounding phase. If all 2-cycles are guaranteed to receive a facility in the random rounding, then all clients will eventually be covered by iteratively following $\phi^{(l)}(i)$ to $\phi^{(l+1)}(i)$ since this sequence eventually reaches the 2-cycle root in $H$.

We now turn our attention to fixing type 1 and type 2 clients. Begin by setting $(\overline{x}^{(4)}, \overline{y}^{(4)}) \leftarrow (\overline{x}^{(3)}, \overline{y}^{(3)})$. Consider any type 1 client $i$ or type 2 client pairs $i$ and $i'$ with contributing facilities $j$ and $j'$. We will build a sequence of locations and facilities starting with $j$. Now, $j$ must be contributing to another location $v$ since it was only contributing $\frac{1}{2}$ to $i$. If there is a type 1 client $i'$, then continue constructing this sequence with $i'$ followed by the other facility $j''$ contributing to $i'$. If there is a type 2 client $i'$, then continue constructing this sequence with $i'$ being followed by $\phi(i')$ and then by the facility $j''$ contributing to $\phi(i')$. Continue to extend the sequence in this way until a location is

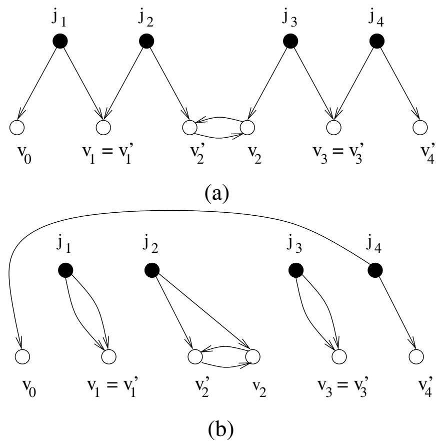



**Figure 2. A sequence of bad clients before (a) and after (b) applying the fixing operation. Each edge represents an assignment of fractional value $\frac{1}{2}$**

reached that is good or until the location of the original type 1 client $i$ is reached. In the former of these two cases, extend the same path from facility $j'$ contributing to the original $i$. This process forms a sequence consisting only of clients of type 1 or 2 (except, perhaps, the endpoints, if we do not get a cycle) and the facilities which contribute to them. If we do this for every type 1 or 2 client, we get a collection of sequences of which each facility $j$ and location $v$ are adjacent in at most one of them (so that each $y_{jv}^{(4)}$ is represented at most once). We will deal with each of these sequences individually.

Say $v_0, j_1, v_1', v_1, j_2, v_2', v_2, \ldots, v_{m-1}', v_{m-1}, j_m, v_m'$ is such a sequence where, for $0 < i < m$, if $v_i$ is a type 1 client we assume $v_i = v_i'$. Finally, if $v_0$ is of type 1 then we have $v_0 = v_m'$ and if $v_0$ is of type 2 then $\phi(v_0) = v_m'$. Since we could consider this sequence in the reverse order, without loss of generality, we can assume that:

$$\sum_{i=1}^{m-1} \left(2d_{j_i v_i'} + d_{j_i v_{i-1}}\right) \leq \sum_{i=1}^{m-1} \left(d_{j_i v_i'} + 2d_{j_i v_{i-1}}\right) \quad (2)$$

Perform the following sequence of updates to fix the type 1 and 2 locations in this sequence (say $v_m = v_0$): $y_{j_i v_{i-1}}^{(4)} \leftarrow 0$ for $1 \leq i \leq m$, and $y_{j_i v_i}^{(4)} \leftarrow y_{j_i v_i}^{(4)} + \frac{1}{2}$, for $1 \leq i \leq m$. Note that by rule 2 above, if $v_i' = v_i$ (i.e. it is a type 1 client) then we are essentially setting $y_{j_i v_i}^{(4)} \leftarrow 1$ and if $v_i' \neq v_i$ we will have $y_{j_i v_i'}^{(4)} = y_{j_i v_i}^{(4)} = \frac{1}{2}$ (see Figure 2 for an example).

There are no more type 1 or type 2 clients remaining left after this update is performed for all of the sequences. It is easy to see that we do not have to change $\overline{x}^{(4)}$ values, therefore since only $\overline{y}^{(4)}$ variables are changed, we have $\overline{C}^{(4)} = \overline{C}^{(3)}$. We can also bound the cost of $\overline{F}^{(4)}$

**Lemma 2.7.** $\overline{F}^{(4)} \leq 4\overline{F}^{(3)}$.

### 2.6 Randomized Rounding

As mentioned before, our final step is to round each facility $j$ to a location $v$ with probability $y_{jv}^{(4)}$. Since for each facility $j$, $\sum_v y_{jv}^{(4)} = 1$, the expected cost of the facilities after rounding is exactly $\overline{F}^{(4)}$. Naturally, for each client that does not have a facility at its location we send it to the nearest location with a facility after this step is performed. We bound the cost increase due to moving clients in the following lemma whose proof is ommitted.

**Lemma 2.8.** *The expected cost of the client assignments after the randomized rounding of $\overline{y}^{(4)}$ is at most $4\overline{C}^{(4)}$.*

### 2.7 Putting it all Together

Working with the modified instance, we have the client/facility costs initially being at most $(\overline{C}, \overline{F})$. After the second step, the new client/facility costs are bounded by $(2\overline{C}, \overline{F} + \overline{C})$. When obtaining the half-integer solution, the costs increase to at most $(2\overline{C} - \tau\Delta, \overline{F} + \overline{C} + \Delta)$ for some constant $\tau$ which we will specify shortly. Fixing type 1 and type 2 clients resulted in the cost of the current solution rising to at most $(2\overline{C} - \tau\Delta, 4\overline{F} + 4\overline{C} + 4\Delta)$. Finally, the random rounding produced an integer solution to the modified instance with an expected cost of at most $(8\overline{C} - 4\tau\Delta, 4\overline{F} + 4\overline{C} + 4\Delta)$.

However, as detailed in the clustering step, we have to move the demands back to their original locations which is done with a penalty of $4\overline{C}$. Thus, the final cost of the algorithm is $16\overline{C} + 4\overline{F} + 4(1-\tau)\Delta$. Choosing the constant $\tau$ to be 1 when obtaining the half-integer solution, we see the overall cost of the final integer solution to the original problem being bounded by $16\overline{C} + 4\overline{F} \leq 16 \cdot OPT_f$.

## 3 Hardness Result

In this section we prove Theorems 1.3 and 1.4.

**Proof of Theorem 1.3:.** Suppose we are given an instance of $k$-median on a graph $G(V, E)$ with metric edge weights $d_{ij}$ and demand $D_v$ for each vertex, and integer $k$. First, using scaling, we assume that the minimum edge length in $G$ is at least 1 and the minimum demand $D_v$ is at least 1. We construct an instance of the mobile facility location problem as follows. Let $\Delta$ denote the maximum distances of this metric and define $\sigma = \alpha n k \Delta$, with $n = |V|$. We use the same graph $G$ and place $k$ facilities in arbitrary nodes of $G$ and let each $v \in V$ be a client with demand $\sigma D_v \geq$

1. Consider any optimum solution of the instance of TM-MFL with cost $\overline{C} + \overline{F}$, where $\overline{C}$ denotes the cost of moving clients and $\overline{F}$ denotes the cost of moving facilities, and any optimum solution with cost $C^*$ to the $k$-median instance.

We claim that $\overline{C} + \overline{F} \leq \sigma C^* + \frac{\sigma}{\alpha n}$. To see this, take the optimum solution of the $k$-median. Moving the demands in TM-MFL as in this solution of $k$-median costs exactly $\sigma C^*$. To bring facilities to these $k$ locations costs at most $k\Delta = \frac{\sigma}{\alpha n}$. Thus we have:

$$\frac{\overline{C} + \overline{F}}{\sigma} \leq C^* + \frac{1}{\alpha n} \leq C^* \left(1 + \frac{1}{\alpha n}\right), \qquad (3)$$

since $C^* \geq 1$.

Now suppose there is an $\alpha$-approximation algorithm for TM-MFL and it returns a solution with cost $C' + F'$. Obtain a solution to the $k$-median based on this approximate solution by moving the demands as in the TM-MFL solution, and let $C''$ be its cost. Using (3): $C'' = \frac{C'}{\sigma} \leq \frac{(C'+F')}{\sigma} \leq \frac{\alpha(\overline{C}+\overline{F})}{\sigma} \leq C^*\alpha\left(1 + \frac{1}{\alpha n}\right)$. Thus $C''$ is within ratio $\alpha + \frac{1}{n}$ of the optimum, i.e. we have an $(\alpha + o(1))$-approximation for $k$-median. □

**Proof of Theorem 1.4:.** NP-completeness of the classic vertex cover problem, proven by Karp [17], is all that is required for this result. Given a graph $G(V, E)$ and an integer $k$, we construct an instance of minimum maximum movement facility location on a new graph $H$ as follows. Let the vertex set of $H$ be $V \cup E \cup \{f_1, \ldots, f_k\}$ where each $f_i$ is a new node. Add an edge from every $f_i$ to every vertex in $V$ with cost 1 and place a facility in each $f_i$. For each $v \in V$ and $e \in E$, if $v$ is an endpoint of $e$ in the original graph $G$ then connect $v$ and $e$ in $H$ with an edge having cost 1. Finally, the set of all clients in this new graph is exactly $E$. It is straightforward to verify that $G$ has a vertex cover of size at most $k$ if and only if $H$ has a solution with maximum movement 1. Consequently, any $(2-\epsilon)$-approximation algorithm, for any $\epsilon > 0$, will return a solution of cost less than 2 if $G$ has a vertex cover of size $k$. Conversely, if $G$ does not have a vertex cover of size $k$ then any algorithm must return a solution of cost 2 in $H$. □

## 4 Concluding Remarks

One natural question is whether we can obtain an approximation algorithm with ratio better than 8 for TM-MFL. Since this generalizes the classical $k$-median problem, improving this ratio beyond 3 would imply an approximation algorithm that is better than the currently best known approximation algorithm for $k$-median.

One direction would be to try local search. As mentioned in the Remark after Theorem 1.1, we can exhibit, for any large enough positive integer $F$, an instance of TM-MFL with locality gap at least $F/(p+2)$ with respect to the two operations defined there, which involves operations involving up to $p$ facilities or locations. To that end, consider a cycle on $F + p + 1$ vertices where all edges have cost 1. Let $v_1, \ldots, v_{F+p+1}$ be the label of the vertices in counter-clockwise order. On vertices $v_i, i = 2 \ldots F - p$, place a client with demand $2i$ and a single facility. On the $2p + 1$ vertices following vertex $F - p$, alternate between placing a client with demand that is 2 more than the previously placed client and placing a facility. Finally, place a single facility on vertex $v_1$. Consider the solution to TM-MFL on this instance where each facility moves counter-clockwise one step. All clients are covered and the total cost of this solution is $F$. One possible way to get to this configuration starting from the initial configuration is by first moving the facility on vertex $F + p$ to location $F + p + 1$ (since this reduces the cost of serving the $2(F + 1)$ facilities there), then moving the facility on vertex $F+p-2$ to location $F+p-1$, and so on. In other words, every facility (starting from the one at location $F + p$ down to the one at location 1) moves one step counter-clockwise to the nearest location that has clients on it. Each of these moves reduces the total cost. The claim is that this solution is a local minimum with respect to the local search operations detailed above. First, since all clients have some facility at their start location and each facility moves only one step, moving any facility from a node with a client to a node without a client will only increase the overall cost. Second, it's not hard to see that permuting the destinations of any $k \leq p$ facilities will not improve the cost. In contrast, a solution of cost $p + 2$ is obtained by moving all facilities that do not start at a client location to the nearest client in the clockwise direction. Therefore, the ratio gap is at least $F/(p + 2)$.

Another direction of research is to consider more general versions of TM-MFL. A natural one is when there is a weight $w_j$ associated with each facility $j$ and the cost of moving this facility to location $i$ is now $w_j d_{ij}$. Our approximation algorithm as it is does not extend to work for this more general setting. An even more general setting is when $w_j$ (for facility $j$) is a concave monoton function and $w_j(i)$ is the cost of moving facility $j$ one unit of distance if the number of demands it services at the end is $i$. However, these more general models seem much more difficult to deal with and getting a constant approximation for any of these seems to require substantially new ideas.

As mentioned in [8], many classical optimization problems can be defined in this movement setting which are both theoretically interesting and have applications in real world. So far there are only a few problems considered in [8] and this paper.

**Acknowledgements:** The second author thanks M. Hajiaghayi for some initial discussions about the problem. The example for local search gap was obtained in some discussions between the second author and Rohit Khandekar, which we thank him.

## References

[1] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit, *Local search heuristics for $k$-median and facility location problems*, SIAM J. Comput. 33 (2004), pp. 544-562.

[2] Y. Bartal, *On approximating arbitrary metrics by tree metrics*, in Proc. of the 29th annual ACM Symp. on Theory of Computing (STOC), 1997, pp. 161–168.

[3] J.L. Bredin, E. Demaine, M.T. Hajiaghayi, and D. Rus, *Deploying sensor networks with guaranteed capacity and fault tolerance*, in Proc. of the 6th annual ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), 2005, pp. 309–319.

[4] Jaroslaw Byrka, *An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem*, In Proceedings of APPROX 2007.

[5] M. Charikar and S. Guha, *Improved combinatorial algorithms for facility location and $k$-median problems*, in Proc. of the 40th Annual IEEE Symp. on Foundations of Comp. Sci. (FOCS), 1999, pp. 378–388.

[6] M. Charikar, S. Guha, D. Shmoys, and E. Tardos, *A constant factor approximation algorithm for the $k$-median problem*, in Proc. of the 31st annual ACM Symp. on Theory of Computing (STOC), 1999, pp. 1–10.

[7] F. Chudak and D.B. Shmoys, *Improved approximation algorithms for uncapacitated facility location problem*, SIAM J. Comput, 33 (2003), pp. 1–25.

[8] E. Demaine, M. Hajiaghayi, H. Mahini, A. Sayid-Roshkar, S. Oveisgharan, and M. Zadimoghaddam, *Minimizing movement*, In Proc. of the 18th annual ACM-SIAM Symp. on Discrete Algorithms (SODA), 2007, pp. 258–267.

[9] N. Devanur, N. Garg, R. Khandekar, V. Pandit, A. Saberi, and V. V. Vazirani, *Price of Anarchy, Locality Gap, and a Network Service Provider Game*, In Proceedings, 1st Workshop on Internet and Network Economics (WINE), Lecture Notes in Computer Science 3828, pages 1046–1055, 2005.

[10] J. Fakcharoenphol, S. Rao, and K. Talwar, *A tight bound on approximating arbitrary metrics by tree metrics*, JCSS, 69 (2004), pp. 485–497. Preliminary version in Proc. 35th Annual ACM Symp. on Theory of Computing (STOC), 2003.

[11] S. Guha and S. Khuller, *Greedy strikes back: improved facility location algorithms*, in Proc. of the 9th annual ACM-SIAM Symp. on Discrete Algorithms (SODA), 1998, pp. 649–657.

[12] D. Hochbaum, *Heuristics for the fixed cost median problem*, Mathematical Programming, 22 (1982), pp. 148–162.

[13] T-R. Hsiang, E.M. Arkin, M. Bender, S. Fekete, and J.S. Mitchell, *Algorithms for rapidly dispersing robot swarms in unknown environments*, in Proc. of the 5th Workshop on Algorithmic Foundations of Robotics (WAFR), 2002, pp. 77–94.

[14] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V.V. Vazirani, *Approximation algorithms for facility location via dual fitting with factor-revealing LP*, J. ACM, 50 (2003), pp. 795–824.

[15] K. Jain, M. Mahdian, and A. Saberi, *A new greedy approach for facility location problems*, in Proc. of the 34th annual ACM Symp. on Theory of Computing (STOC), 2002, pp. 731–740.

[16] K. Jain and V.V. Vazirani, *Approximation algorithms for metric facility location and kmedian problems using the primal-dual schema and Lagrangian relaxation*, J. ACM, 48 (2001), pp. 274–296.

[17] R. Karp, *Reducibility among combinatorial problems*, In Raymond E. Miller and James W. Thatcher, editors *Complexity of Computer Computations*, 85–103, Plenum Press, 1972.

[18] M. Mahdian and M. Pál, *Universal facility location*, in Proc. of the 11th annual European Symp. on Algorithms (ESA), 2003, pp. 409–421.

[19] M. Mahdian, Y. Ye, and J. Zhang, *Approximation algorithms for metric facility location problems*, SIAM J. Comput., 36 (2006), pp. 411–432.

[20] A. Schrijver, Combinatorial Optimization - Polyhedra and Efficiency, Springer-Verlag, New York, 2005.

[21] D. Shmoys, É. Tardos, and K. Aardal, *Approximation algorithms for facility location problems*, in Proc. of the 29th annual ACM Symp. on Theory of Computing (STOC), 1997, pp. 265–274.

[22] M. Sviridenko, *An improved approximation algorithm for the metric uncapacitated facility location problem*, in Proc. of 9th Inter. Conf. on Integer Programming and Combinatoril Optimization (IPCO) 2002, pp. 240–257.