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 facility location, 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 [11] where a simple 2-approximation was proposed for the minimum maximum movement mobile facility location problem 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. Our algorithm is obtained by rounding an LP relaxation in five phases. 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 proposed in [11] 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 their total demand into one big shipment to be sent from the plant to the distribution center. This problem can be modeled using the following natural generalization of the classical k-median and variation of the facility location problem. 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 $i \in C$ located at a node (these correspond to retailers). To handle multiple clients at a single location, we assume each location has at most one client and that each client $i \in C$ has demand D_i. Therefore, we can view C as a subset of V and we think of D_i as being the number of clients initially located at i. This also allows an efficient representation of instances where the number of clients is exponentially larger than the number of

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

nodes. 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 $\sum_{j \in F} d_{jv_j} + \sum_{i \in C} D_i d_{iv_i}$ 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 the *minimum maximum movement mobile facility location* problem, 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 the demand (number of individual clients) per node is irrelevant in MM-MFL since we are only concerned with the distance.

These problems fall into a natural class of problems, called movement problems, which were introduced by Demaine et al. [11]. In these types of problems, we are typically 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 (in which our desired property P is that the subgraph induced by the final pebbles' locations is connected), s, t-connectivity (in which two given nodes s and t have to be in the same component in the subgraph induced by the final pebbles' locations), and independent set in $\mathbb{R}^2$ (the Euclidean distance between every pair of pebbles should be at least 1). 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 function. Some of the natural objective functions considered are the total distance traveled by all pebbles or the maximum distance a pebble has to move (distance in a graph is the shortest path and $\mathbb{R}^2$ distance is Euclidean distance). 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. [16, 6] and the references in [11] for more applications).

Demaine et al. [11] give approximation algorithms and hardness results (for different objective functions) for the properties P mentioned earlier. They also raise the question of minimizing movement in mobile facility location problems. For the minimum maximum movement mobile facility location problem (MM-MFL), they [11] 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 problem (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 problem 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 the minimum total movement facility location problem.

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 [15]. Shmoys, Tardos, Aardal [24] 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 [14, 10, 17, 19, 3, 25, 22, 7] and references in [7]). The best known algorithm has ratio 1.5 [7]. Guha and Khuller [14] 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 the capacitated facility location problem, in which there is a capacity on the number of clients that can be served at each facility f_i (e.g.

see [21] and the references there).

The TM-MFL problem can be thought of as a variant of the classical facility location problem. In both problems, the goal is to assign clients to facilities to minimize the total cost. The ideas of moving a client (as in TM-MFL) versus assigning clients (as in classical facility location) can be thought of as equivalent. Similarly, the cost of moving a facility in TM-MFL is also analogous, but not equivalent, to the opening cost of a facility in classical facility location. These ideas are not equivalent since in TM-MFL, although there are $|V|$ potential locations for facilities, we can only open at most $|F|$ out of the $|V|$ many locations for facilities and the cost of opening a facility at a certain location $v \in V$ is dependent on the other at most $|F| - 1$ locations that are opened. That is, the cost of assigning facilities some set $U \subseteq V, |U| \leq |F|$ locations is determined by a minimum cost matching between the initial locations of facilities and the destinations in U.

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 [3]. On the other hand, Jain, Mahdian, and Saberi [18] 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. TM-MFL can be thought of as a variant of the k-median problem in that we have a bounded number of locations we can select to serve clients. However, in addition to minimizing the cost of serving clients, we must consider the cost of assigning (*i.e.* moving) facilities to these locations.

Other variants and generalizations of the uncapacitated facility location and k-median problems have been studied. One popular variant is the k-facility location problem which is the classical uncapacitated facility location problem with the additional restriction that at most k facilities be opened. Like k-median, the current best approximation ratio $(2+\sqrt{3}+\epsilon)$ for k-facility location is obtained through a local search algorithm [26]. Initially, it might appear that the k-facility location problem seems similar enough to our problem that it may be possible to model TM-MFL as a k-facility location problem by somehow having the opening costs in k-facility location correspond to movement costs in TM-MFL. However, we show that a local search technique for TM-MFL that is analogous to the local search for k-facility location has an unbounded locality gap. This highlights the fact that allowing facilities to be mobile creates a fundamentally different problem.

The mobile facility location problem is, in some ways, reminiscent of the k-level facility location problem with $k = 2$. Here, we have k sets of facilities $F_1, \ldots, F_k$ where each facility has an opening cost and each client i is assigned a sequence of k facilities $f_{i1}, \ldots, f_{ik}$ where $f_{ij} \in F_j$. Say client i starts at location v, then the service cost of client i is $d_{vf_{i1}} + d_{f_{i1}f_{i}2} + \ldots + d_{f_{i(k-1)}f_{ik}}$. The goal is to minimize the total service cost of all clients plus the opening cost of all facilities that are serving at least one client. If the distances are metric then the k-level facility location problem admits a 3-approximation [1] while the special case of $k = 2$ has a 1.77-approximation [27]. To some it might seem that our problem can be modeled somehow as a 2-level facility location problem; one level could be the clients moving to their destination and the other level could be moving from these destinations to the original facility assigned to that particular destination. However, there are several fundamental differences between the 2-level facility location problem and TM-MFL. For example, if we move clients from level 1 to level 2 as described above, then the distance from a level 1“facility” to the level 2 would be counted toward the total movement cost (as many times as there are clients traveling this route) whereas in TM-MFL each facility is moved from level 2 to level 1 and this distance is only counted once toward the total cost of the solution. Also, while there are n potential facilities in level 1, we can use at most $|F|$ such locations and no two clients visiting different level 1 facilities could visit the same level 2 facility since a mobile facility can only move to one location. Finally, 2-level facility location has a 1.77-approximation whereas we show that TM-MFL cannot be approximated within any constant better than k-median, for which only the best approximation achieves a factor of $3 + \epsilon$. So solving TM-MFL by modeling it as a 2-level facility location instance seems difficult.

Demaine et al. [11] 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, 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 $\mathbb{R}^2$).

Finally, we mention that the name "mobile facility location" has been used to refer to the study of how facilities move to optimally serve a set of clients who are moving in a continuous manner in $\mathbb{R}^d$ (see, *e.g.* [5]). This problem seems mostly unrelated to the problem discussed in this paper.

Our results: We consider both the TM-MFL and MM-MFL problems. For TM-MFL restricted to trees it is possible to obtain a pseudo-polynomial time exact algorithm using dynamic programming 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 $O(\log n)$ [4, 12], this yields an $O(\log n)$-approximation for TM-MFL in general graphs in pseudo-polynomial time [13]; 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 the minimum total movement mobile facility location problem (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. [9] 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. 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 [14]:

Theorem 1.2 *The minimum total movement mobile facility location problem (TM-MFL) 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 $3 + \epsilon$ [3].

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

Jain, Mahdian, and Saberi [18] 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 [11]) 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 (this was independently discovered by Armon [2]):

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 [3] 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 any natural local search algorithm that performs a bounded number of exchange or switch operations at each iteration will have an unbounded ratio. 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 the following local search operation; select a subset of $k \leq p$ facilities $f_{i_1}, \ldots, f_{i_k}$ and a subset of k destinations $v_1, \ldots, v_k$ and move f_{i_j} to v_j for each $1 \leq j \leq k$. Note that this operation allows to permute the final location of facilities among themselves too. Consider an algorithm that iteratively performs such local search operations that improve the overall cost of the solution until no more improvements can be made. In section 4, we show there are examples for which this algorithm will have unbounded approximation ratio.

The rest of the paper is organized as follows. For ease of exposition we start by presening a proof for a slightly weaker version of Theorem 1.1 in Section 2. Then in Section 3 we show how to improve this algorithm to obtain the result of Theorem 1.1. Section 5 contains the proofs of hardness results. We conclude the paper with a few remarks.

2 A Randomized 16-Approximation Algorithm

In this section, we present a randomized 16-approximation algorithm for the minimum total movement mobile facility location problem. This algorithm uses randomized rounding of the optimal fractional solution obtained from solving a natural IP/LP formulation. In the next section we show how we can derandomize this algorithm as well as improve the analysis to obtain a deterministic 8-approximation algorithm.

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, 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:

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

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 $d_{ii'} > 4 \cdot \max\left\{\overline{C}_i, \overline{C}_{i'}\right\}$. 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: We first describe the final step to provide motivation for the fourth step. The final step obtains an integer solution by first fixing a destination for each facility and then assigning each client to a covered location. This step first has each facility being uniformly randomly rounded to one of its at most 2 (fractional) destinations. Then we move each client i which is not covered to $\phi(i)$ and from there to $\phi(\phi(i))$ if there is no facility assigned there either, and so on, until the client reaches a covered location. The expected facility cost is exactly the current fractional cost. The expected cost of moving the clients will be bound by moving each client from i to $\phi(i)$, iteratively.

In order to bound the cost of moving clients in the final step, the fourth step will prepare the current half-integer solution for the randomized rounding step. There are two types of clients that we will consider as *"bad"*. The first type consists of clients i with $x_{ii}^{(4)} = 1$ while having two distinct facilities j and j' such that $y_{ji}^{(4)} = y_{j'i}^{(4)} = \frac{1}{2}$. This is bad because the client currently has cost $\overline{C}_i = 0$ (it is completely covered at its own location), but randomly rounding facilities based on their $y_{jv}^{(4)}$ values may result in location i not being covered; therefore we cannot bound the increase in the cost of that client. The second type of bad client is one with $\phi(\phi(i)) = i$, but both i and $\phi(i)$ have their locations covered with weight $\frac{1}{2}$ by different facilities. This is bad since the event where neither i nor $\phi(i)$ are covered results in client i being moved between i and $\phi(i)$ back and forth. Notice client i being bad in this case implies $\phi(i)$ is also bad. Through this step, we ensure that there is no bad client (or client pairs) and so we are able to say that every client must be eventually covered as it follows $\phi(i)$ and the cost of moving each client increases, in expectation, by a constant factor of its original half-integer 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 [9]. 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**
    let i' be any such client
    $D_{i'} \leftarrow D_{i'} + D_i$
    $D_i \leftarrow 0$ (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 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 $T + 4\overline{C}$.*

Proof. Simply move the location of each unit of demand in the modified instance to the location of that unit of demand in the original instance. By the definition of the new instance, the cost increase for the unit of demand due to this move is bound by $4\overline{C}_i$ where i was the original client that possessed that demand. Summing over all units of demand and original clients i, the cost increase is at most $4\overline{C}$. □

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

Proof. Consider each unit of demand of a client. The contribution of that unit of demand to $\overline{C}^{(1)}$ is exactly the same as in $\overline{C}$ if the demand is not moved. If the demand is moved, then it moves to a new location with a smaller index which implies a reduction in cost since the $\overline{C}_i$ are considered in increasing order. The cost of the facilities stays the same since their assignments were not changed. □

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:

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

where the first inequality uses the fact that $d_{ji'} \leq d_{jv} + d_{vi'}$ by triangle inequality. 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 bound 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.*

Proof. This follows from a simple averaging argument. Consider a client i and the set of all vertices v with distance $d_{iv} \leq 2\overline{C}_i^{(1)}$; we call this a ball with radius $2\overline{C}_i^{(1)}$ around i, denoted by R_i. First, we claim that for every pair i, i' of clients, R_i and $R_{i'}$ are disjoint. Otherwise, if v belongs to both R_i and $R_{i'}$, with $\overline{C}_{i'}^{(1)} < \overline{C}_i^{(1)}$, then we have $d_{i'v} \leq 2\overline{C}_i^{(1)}$ which implies $d_{iv} + d_{i'v} \leq 4\overline{C}_i^{(1)}$. By the triangle inequality, this results in $d_{ii'} \leq 4\overline{C}_i^{(1)}$ which contradicts Lemma 2.3. Next we claim that $\sum_{v \in R_i} x_{iv}^{(1)} \geq \frac{1}{2}$, for otherwise, the total cost for fractional values assigned to the vertices outside R_i is strictly larger than $\overline{C}_i^{(1)}$, a contradiction. Since all client assignments to the vertices v inside R_i are moved to i then $x_{ii}^{(2)} \geq \frac{1}{2}$.

$\square$

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 $\sum_j y^{(2)}_{jv} > 0$. Then, for each client i if $\sum_j y^{(2)}_{ji} > 1$ we can move coverage from facilities from $y^{(2)}_{ji}$ to $y^{(2)}_{jj}$ until $\sum_j y^{(2)}_{ji} = 1$ 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, $\sum_j y^{(2)}_{ji} \leq 1$ for all locations v and, for all clients i, $x^{(2)}_{ii} \geq \frac{1}{2}$ and $\sum_{i' \in C} x^{(2)}_{ii'} = 1$. We can assume that the remaining $1 - x^{(2)}_{ii} \leq \frac{1}{2}$ 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 $x^{(2)}_{ii} = \sum_j y^{(2)}_{ji}$ by moving $\sum_j y^{(2)}_{ji} - x^{(2)}_{ii}$ from $x^{(2)}_{i\phi(i)}$ to $x^{(2)}_{ii}$ at no additional cost.

2.4 Getting a Half-Integer Solution

In this phase our goal is to ensure that the value of each $x^{(3)}_{iv}$ and $y^{(3)}_{jv}$ 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 $\sum_j y^{(3)}_{jv} \in \{0, \frac{1}{2}, 1\}$. 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 $y^{(3)}_{jv}$ 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.

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 α_k for v'. These α_0, α_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 τ 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:

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

What we plan to do is shift some coverage from v to v' through this path, by simultaneously increasing each $y^{(3)}_{j_m v_m}$ and decreasing each $y^{(3)}_{j_m v_{m-1}}$ (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 $\epsilon = \min\{y^{(3)}_{j_m, v_{m-1}} | 1 \leq m \leq k\}$. If v is a client i, then update $\epsilon \leftarrow \min\{\epsilon, x^{(3)}_{ii} - \frac{1}{2}\}$. Finally, update $\epsilon \leftarrow \min\{\epsilon, 1 - \sum_j y^{(3)}_{jv'}\}$ (this is $1 - x^{(3)}_{i'i'}$ 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 δ. 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 δ. Letting Δ be the sum of the δ values over all executions of such an update, we see that $\overline{C}^{(3)} \leq \overline{C}^{(2)} - \tau\Delta$ and $\overline{F}^{(3)} = \overline{F}^{(2)} + \Delta$.

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

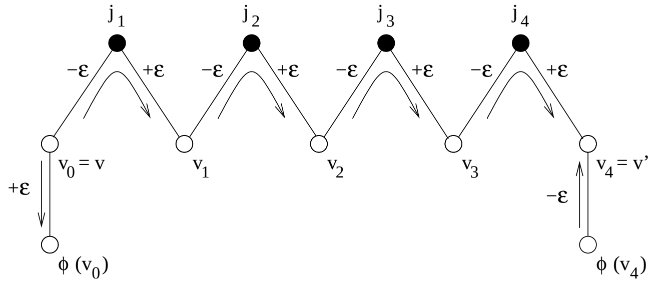

Figure 1: Changing fractional values over a path between two clients with non-half-integer coverage.

The previous process obtains a half-integer solution in that each location v has $\sum_j y^{(3)}_{jv} \in \left\{0, \frac{1}{2}, 1\right\}$. However, it is not necessarily true that each $y^{(3)}_{jv}$ 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. [23]). 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 $y'_{jv} \in \left\{0, \frac{1}{2}, 1\right\}$. 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 $y^{(3)}_{jv}$ is half-integer, for all j, v. Since each client i has $x^{(3)}_{ii} \ge \frac{1}{2}$ and uses its own coverage to the maximum amount, it follows that each $x^{(3)}_{ii} \in \{\frac{1}{2}, 1\}$ and if $x^{(3)}_{ii} = \frac{1}{2}$ then $x^{(3)}_{i\phi(i)} = \frac{1}{2}$.

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 ϕ, 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 $x^{(3)}_{ii} = 1$ and for two distinct facilities j and j', $y^{(3)}_{ji} = y^{(3)}_{j'i} = \frac{1}{2}$. Also, a pair of clients i and i' is called *type 2* if $\phi(i) = i'$, $\phi(i') = i$ and $x^{(3)}_{ii} = x^{(3)}_{i'i'} = \frac{1}{2}$ but two different facilities j and j' are such that $y^{(3)}_{ji} = y^{(3)}_{j'i'} = \frac{1}{2}$. The remaining locations are *good* ones. Recall that the final step of the algorithm will round each facility j to location v with probability $y^{(4)}_{jv}$. With this in consideration, we see that type 1 clients are those where $\overline{C}^{(3)}_i = 0$ 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 randomized 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' at v, then continue constructing this sequence with i' followed by the other facility j'' contributing to i'. If there is a type 2 client i' at v, 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 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 $y^{(4)}_{jv}$ is represented at most once). Refer to figure 2(a) for an example of such a sequence. Notice that each type 1 client and type 2 pair of clients appears exactly once in exactly one such sequence (it may be that a client appears at the start and end of a sequence; we can think of this sequence as a cycle in which the client appears 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:

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

Perform the following sequence of updates to fix the type 1 and 2 locations in this sequence (say $v_{m'} = v_0$): $y^{(4)}_{j_i v_{i-1}} \leftarrow 0$ for $1 \leq i \leq m$, and $y^{(4)}_{j_i v_i} \leftarrow y^{(4)}_{j_i v_i} + \frac{1}{2}$, 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 $y^{(4)}_{j_i v_i} \leftarrow 1$ and if $v'_i \neq v_i$ we will have $y^{(4)}_{j_i v'_i} = y^{(4)}_{j_i v_i} = \frac{1}{2}$ (see Figure 2 for an example).

There are no more type 1 or type 2 clients remaining 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 $\overline{F}^{(4)}$ as follows:

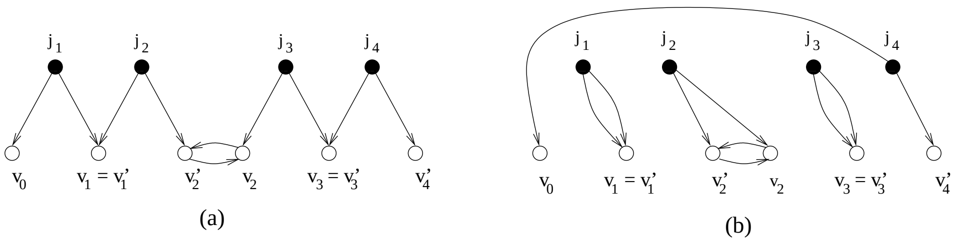

Figure 2: A sequence of bad clients before (a) and after (b) applying the fixing operation. Each edge represents an assignment of fractional value $\frac{1}{2}$

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

Proof. The overall increase in facility costs for a given $v_0, j_1, v'_1, v_1, j_2, \ldots, v'_{m-1}, v_{m-1}, j_m, v'_m$ sequence is given by:

$$\begin{aligned}
\frac{1}{2}\left(\sum_{i=1}^{m-1}(d_{j_i v_i} - d_{j_i v_{i-1}}) + d_{j_m v_0} - d_{j_m v_{m-1}}\right) &\leq \frac{1}{2}\left(\sum_{i=1}^{m-1}(d_{j_i v_i} - d_{j_i v_{i-1}}) + \right. \\
&\qquad \left. \sum_{i=1}^{m-1}(d_{j_i v_{i-1}} + d_{j_i v_i}) + d_{j_m v_{m-1}} - d_{j_m v_{m-1}}\right) \\
&= \sum_{i=1}^{m-1} d_{j_i v_i} \\
&\leq \sum_{i=1}^{m-1}(d_{j_i v'_i} + d_{v'_i v_i}) \\
&\leq \sum_{i=1}^{m-1}(d_{j_i v'_i} + d_{v'_i v_{i-1}}) \\
&\leq \sum_{i=1}^{m-1}(d_{j_i v'_i} + d_{j_i v'_i} + d_{j_i v_{i-1}}) \\
&\leq \frac{3}{2}\left(\sum_{i=1}^{m-1} d_{j_i v'_i} + d_{j_i v_{i-1}}\right) \\
&= 3\sum_{i=1}^{m-1} \overline{F}^{(3)}_{j_i}
\end{aligned}$$

The first, second and fourth inequalities come from the triangle inequality. For each $0 < i < m$, if v'_i is of type 1, then $d_{v'_i v_i} = 0$ and if v'_i and v_i are of type 2, then since $\phi(v'_i) = v_i$ we have that $d_{v'_i v_i} \leq d_{v'_i v_{i-1}}$. In either case, the third inequality holds. The last inequality comes from the assumption in expression (2). Summing over all such paths and cycles yields $\overline{F}^{(4)} \leq 4\overline{F}^{(3)}$ because each $y^{(4)}_{jv}$ variable was updated at most once. □

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. Recall that $\phi^{(k)}(i)$ is defined recursively as $\phi^{(0)}(i) = i$ and $\phi^{(k+1)}(i) = \phi(\phi^{(k)}(i))$ for $k \geq 0$.

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

Proof. For any client i with $x_{ii}^{(4)} = 1$ we have $\overline{C}_i^{(4)} = 0$. Since there are no type 1 clients, then location i is guaranteed to receive a facility so the new cost of this client is still 0. Now assume that client i has $x_{ii}^{(4)} = \frac{1}{2}$. We now prove, for any $k \geq 1$, that:

$\Pr(\phi^{(k)}(i)$ does not receive a facility $\mid \phi^{(l)}(i)$ did not receive a facility for all $0 \leq l < k) \leq \frac{1}{2}$.

Since there are no type 1 clients, if $x_{\phi^{(k)}(i)\phi^{(k)}(i)}^{(4)} = 1$ then $\phi^{(k)}(i)$ will receive a facility. Otherwise, let j be the facility with $y_{j\phi^{(k)}(i)}^{(4)} = \frac{1}{2}$. If $y_{j\phi^{(l)}(i)}^{(4)} = 0$ for all $0 \leq l < k$ (that is, j is not partially assigned to any location before the k'th step along ϕ starting at i), then the probability that j will be assigned to $\phi^{(k)}(i)$ is exactly $\frac{1}{2}$. If $y_{j\phi^{(l)}(i)}^{(4)} = y_{j\phi^{(k)}(i)}^{(4)} = \frac{1}{2}$ for some $0 \leq l < k$, then it must be that $\phi^{(l)}(i) \neq \phi^{(k)}(i)$. If this were not true, then this implies that $\phi^{(l)}(i)$ is in a cycle of the graph H of the ϕ function considered in the previous step. Since all cycles of H have length 2 and since there are no type 2 clients, then $\phi^{(l)}(i)$ not receiving a client implies $\phi^{(l+1)}(i)$ must have received one. Since we assume that all $\phi^{(l')}(i)$ do not receive a facility for $0 \leq l' < k$, then it must be that $l + 1 = k$ which implies the contradiction $\phi^{(l)}(i) = \phi^{(k)}(i) = \phi^{(l+1)}(i)$. Therefore, since $\phi^{(l)} \neq \phi^{(k)}(i)$ and $\phi^{(l)}(i)$ does not receive facility j then j must have been assigned to location $\phi^{(k)}(i)$. Therefore, in each possible case $\phi^{(k)}(i)$ receives a facility with probability at least $\frac{1}{2}$ which proves our claim.

From this we see:

$$\begin{aligned} & \Pr(\phi^{(l)}(i) \text{ do not receive a facility for all } 0 \leq l < k) \\ = & \prod_{l'=0}^{k-1} \Pr(\phi^{(l')}(i) \text{ does not receive a facility} \mid \phi^{(l)}(i) \text{ did not receive a facility for all } 0 \leq l < l') \\ \leq & 2^{-k}. \end{aligned}$$

Note that because we do not have a type 2 client pair, for every pair of clients i, i' with $\phi(i) = i'$ and $\phi(i') = i$ (i.e. they form a 2-cycle in H), there must be one facility j with $y_{ji'}^{(4)} = y_{ji}^{(4)} = \frac{1}{2}$. So exactly one of i or i' will be covered by j after the rounding. Since any walk on H eventually leads to a 2-cycle (containing the root) of which one location must be assigned a facility, then there is a minimum value k_i such that $\phi^{(k_i)}(i)$ will receive a facility with probability 1 if the previous locations $\phi^{(l)}(i)$ do not receive a facility for $0 \leq l < k_i$.

Also, since the weights of edges of H do not increase in any walk, then it is easy to prove by induction that $d_{i\phi^{(k)}(i)} \leq k \cdot d_{i\phi(i)}$. Thus, the expected cost of serving client i with $x_{ii}^{(4)} = \frac{1}{2}$ is bound by:

$$\begin{aligned}
& \sum_{k=0}^{k_i} D_i \cdot d_{i\phi^{(k)}(i)} \cdot \Pr(\phi^{(k)}(i) \text{ receives a location and } \phi^{(l)}(i) \text{ do not for all } 0 \le l < k) \\
\le \; & \sum_{k=0}^{k_i} k \cdot D_i \cdot d_{i\phi(i)} \cdot \Pr(\phi^{(l)}(i) \text{ do not receive a facility for all } 0 \le l < k) \\
\le \; & D_i \cdot d_{i\phi(i)} \cdot \sum_{k=0}^{\infty} \frac{k}{2^k} = 2 \cdot D_i \cdot d_{i\phi(i)} = 4\overline{C}_i^{(4)} D_i,
\end{aligned}$$

and so the lemma follows. □

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 τ 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 τ to be 1 when obtaining the half-integer solution, we see the overall cost of the final integer solution to the original problem being bound by $16\overline{C} + 4\overline{F} \le 16 \cdot OPT_f$.

3 Derandomizing and Improving to an 8-Approximation

In this section we show how we can build upon the algorithm of Section 2 to prove Theorem 1.1. The 8-approximation algorithm is essentially the same algorithm as presented. First, we show how we can derandomize that algorithm using the method of conditional expectations to get a deterministic one. Next we describe a more careful analysis of some of the steps in the algorithm which yields an improved ratio of 8 for the approximation. We will not fix the value of τ until the end of this new analysis.

3.1 Derandomizing the Algorithm of Theorem 1.1

As said before, we use the method of conditional expectations to do the final step of our rounding algorithm deterministically. Since we will move each client i to its nearest location that received a facility after the rounding, we can efficiently compute the expected cost of client i given that some of the facilities are already rounded in the following manner. Begin by ordering the locations $v_1, v_2, \ldots, v_{|V|}$ so that $d_{iv_1} \le d_{iv_2} \le \cdots \le d_{iv_{|V|}}$. The expected cost can be expressed as:

$$\sum_{m=1}^{|V|} d_{iv_m} \Pr(v_m \text{ receives a facility and } v_l \text{ do not for } 1 \le l < m)$$

We can compute the expected cost for client i in the following manner:

Cost(i, m)

if there is a facility assigned to v_m with weight 1 **then** return d_{iv_m}

else if there is a facility j that is assigned to v_m with weight $\frac{1}{2}$ **then**

  if j is also assigned to v_l with weight $\frac{1}{2}$ for some $l < m$, **then** return d_{iv_m}

  else return $\frac{1}{2} \cdot d_{iv_m} + \frac{1}{2} \cdot$ Cost $(i, m+1)$

else return Cost$(i, m+1)$

Lemma 3.1 *The expected cost of client i for a given partial assignment of facilities is Cost$(i, 0, \emptyset)$.*

Proof. First of all, the recursive routine reaches a base case since all facilities are assigned with weights 1 or $\frac{1}{2}$ to locations in $|V|$. Say this base case happens when considering location v_m. We prove, in an inductive fashion, that the value returned when considering location v_l for $1 \leq l \leq m$ is the expected assignment cost of client i given that all locations $v_{l'}, 1 \leq l' < l$ are not allocated a facility.

Consider the base case of v_m. If v_m has a weight 1 assignment, then since there are no type 1 clients and all unused assignments do not move from their initial starting position it must be that some facility is completely assigned to v_m. If v_m has a weight $\frac{1}{2}$ assignment, then since this is a base case it must be that the facility j partially assigned to v_m was also partially assigned to some $v_{l'}$ for $1 \leq l' < m$. Since j was not assigned to $v_{l'}$ by assumption, it must be assigned to v_m. In either case, the value returned is d_{iv_m}.

Now consider some v_l locations for $1 \leq l < m$. The first and second cases of the recursive function do not apply because v_l is not a base case. If there is no facility that is partially assigned to v_l then the probability of v_l receiving a facility is 0. Thus, the expected cost is exactly the expected cost of v_{l+1} receiving a facility given that no $v_{l'}$ receives a facility for any $1 \leq l' \leq l$. If v_l was partially assigned a facility j, then it must be that j is not partially assigned to a previous location $v_{l'}$ for $1 \leq l' < l$ (otherwise v_l is a base case). Thus, the probability of j being assigned to v_l is exactly $\frac{1}{2}$ and the assignment cost for i is d_{iv_l} in this case. The probability that j is not assigned to v_l is also $\frac{1}{2}$. In this event, the expected assignment cost of i is then recursively computed. □

Now consider each facility j in some order. If j is such that $y^{(4)}_{jv} = 1$ for some v, then we assign j to this location. Otherwise, say v and v' are the two locations such that $y^{(4)}_{jv} = y^{(4)}_{jv'} = \frac{1}{2}$. Try the two cases of assigning j to v and v' and pick the one that produces the least expected cost. When this is done for all facilities, we have an integer assignment of facilities and clients to nodes whose cost is at most the expected value before any of this rounding was performed.

3.2 Improving the Relocation of Coverage

Consider the second step of the algorithm which consists of relocating the facilities (and the clients with them). Previously, when we move facility coverage from a location v to the nearest client i', we also moved all client assignments from v to client i'. Instead of doing that, we will move only $x^{(2)}_{i'v}$ from each client (assigned to v) and leave any extra client assignment at v. That is, for each client i let $\epsilon_i = \min\left\{x^{(2)}_{iv}, x^{(2)}_{i'v}\right\}$ and move this much assignment from v to i'. Define $M = \frac{x^{(2)}_{i'v}}{\sum_j y^{(2)}_{jv}}$ and move this fraction of all facility assignments from v to i. In other words, for each client i and facility j, perform the following update:

$$\begin{aligned}
x^{(2)}_{ii'} &\leftarrow x^{(2)}_{ii'} + \epsilon_i \\
x^{(2)}_{iv} &\leftarrow x^{(2)}_{iv} - \epsilon_i \\
y^{(2)}_{ji'} &\leftarrow y^{(2)}_{ii'} + M \cdot y^{(2)}_{jv} \\
y^{(2)}_{jv} &\leftarrow (1 - M) \cdot y^{(2)}_{jv}
\end{aligned}$$

It may be that after such an update that there are still clients assigned to location v. This happens in the case that $x^{(2)}_{i'v} < \max\left\{x^{(2)}_{iv} \mid i \in C\right\}$. However, we have moved exactly $x^{(2)}_{iv}$ total facility assignments from v to i, so the clients with some assignment to i or v are still covered and client i' now has $x^{(2)}_{i'v} = 0$. We repeat this process on vertex v with the next closest client i'' that still has $x^{(2)}_{i''v} > 0$ after the re-assignment. Iterate this process until all client assignments to v have been moved to some client location. For each location v and client i', where the assignments to v were (partially) moved to i' during this process, let $\alpha_{v,i'}$ denote the cost of moving the fraction of assignment of i' to v back to i'; this is $D_{i'}$ times $d_{i'v}$ times the value of $x^{(2)}_{i'v}$ at the time of this movement (note that the client cost for i' decreases by $\alpha_{v,i'}$). Let α_v be the sum of all $\alpha_{v,i'}$ over all iterations of the above procedure on v. If we consider each client i' that had some portion of the assignment $x^{(2)}_{i'v}$ moved back to $x^{(2)}_{i'i'}$, then the total cost of these clients decreases by α_v and the total facility cost increases by at most α_v. Denote by β_v the cost that all other clients i paid to have their assignments moved from v to some i' location where $i \neq i'$. Let $\alpha = \sum_v \alpha_v$ and $\beta = \sum_v \beta_v$ and notice that $\alpha + \beta \leq \overline{C}^{(1)}$ since each fraction of each client is moved at most once. Therefore, the new facility cost $\overline{F}^{(2)}$ is at most $\overline{F}^{(1)} + \alpha$ and the new client cost $\overline{C}^{(2)}$ is at most $\overline{C}^{(1)} + \beta - \alpha$.

3.3 Improving the Analysis of the Expected Cost

In the randomized rounding phase, as observed before, for each client i there is a minimum value k_i such that

$$\Pr(\phi^{(k_i)}(i) \text{ receives a facility} \mid \phi^{(l)}(i) \text{ do not receive a facility } 0 \leq l < k_i) = 1.$$

For each $0 \leq l < k_i$, the probability of $\phi^{(l)}(i)$ receiving a facility given that $\phi(l')(i)$ do not for all $0 \leq l' < l$ is exactly $\frac{1}{2}$. Therefore, the expected cost for client i after the randomized rounding of facilities can be bound as follows:

$$\begin{aligned}
& \sum_{k=0}^{k_i} D_i \cdot d_{i\phi^{(k)}(i)} \cdot \Pr(\phi^{(k)}(i) \text{ receives a location and } \phi^{(l)}(i) \text{ do not for all } 0 \leq l < k) \\
= \; & \sum_{k=0}^{k_i} D_i \cdot d_{i\phi^{k}(i)} \cdot \Pr(\phi^{(l)}(i) \text{ do not receive a facility for all } 0 \leq l < k) \\
& \qquad \cdot \Pr(\phi^{(k)}(i) \text{ receives a location} \mid \phi^{(l)}(i) \text{ does not for all } 0 \leq l < k) \\
= \; & \frac{D_i \cdot d_{i\phi^{(k_i)}(i)}}{2^{k_i}} + D_i \sum_{k=0}^{k_i-1} \frac{d_{i\phi^{(k)}(i)}}{2^{k+1}} \\
\leq \; & \frac{D_i \cdot k_i \cdot d_{i\phi(i)}}{2^{k_i}} + D_i \sum_{k=0}^{k_i-1} \frac{k \cdot d_{i\phi(i)}}{2^{k+1}} \\
= \; & D_i \cdot d_{i\phi(i)} \cdot \left(\frac{k_i}{2^{k_i}} + 1 - \frac{k_i+1}{2^{k_i}}\right) \\
= \; & D_i \cdot d_{i\phi(i)} \cdot \left(1 - \frac{1}{2^{k_i}}\right),
\end{aligned}$$

where the second last equality can easily be verified by induction. This shows that the cost increase for each client i is at most $\overline{C}^{(4)}_i$ (note that the final expression is 0 if $k_i = 0$).

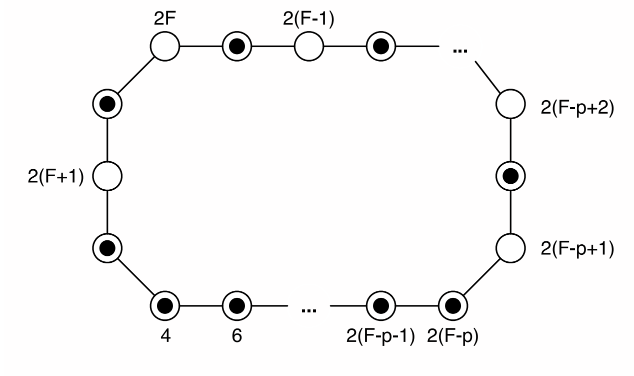

Figure 3: An example with a large locality gap. A dot represents a facility and a number represents client demand.

3.4 The Final Analysis

As before, we start with the modified instance which has client/facility costs at most $(\overline{C}, \overline{F})$. The new analysis on the relocating step shows that the cost can be then be bound by $(\overline{C}+\beta-\alpha, \overline{F}+\alpha)$ where $\alpha+\beta \leq \overline{C}^{(1)} \leq \overline{C}$. The cost after obtaining the half integer solution increases to at most $(\overline{C}+\beta-\alpha-\tau\Delta, \overline{F}+\alpha+\Delta)$. Fixing type 1 and type 2 clients then implies the resulting costs increase to at most $(\overline{C}+\beta-\alpha-\tau\Delta, 4\overline{F}+4\alpha+4\Delta)$. Finally, the new analysis on the expected cost increase, along with the derandomization, shows the integer solution to the modified instance has client/facility costs at most $(2\overline{C}+2\beta-2\alpha-2\tau\Delta, 4\overline{F}+4\alpha+4\Delta)$.

As before, move the demands back to their original locations with a cost increase of $4\overline{C}$. By choosing $\tau = 2$, we have the final cost of the integer solution being bound by:

$$\begin{aligned} 6\overline{C}+4\overline{F}+2\alpha+2\beta+(4-2\tau)\Delta &\leq 8\overline{C}+4\overline{F}+(4-2\tau)\Delta \\ &= 8\overline{C}+4\overline{F} \\ &\leq 8 \cdot OPT_f \end{aligned}$$

4 Instances With Large Locality Gap

Let p be a fixed positive integer and consider the following local search operation. Select a subset $k \leq p$ 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 $1 \leq j \leq k$. Finish by reassigning clients to their nearest facility.

We will 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 above operations. 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. This instance is illustrated in Figure 3.

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. Consider some local search operation that moves a set of $k \leq p$ facilities $f_{i_1}, \ldots, f_{i_k}$ to locations $v_1, \ldots, v_k$. Say, wlog, that c is the index such that all $v_i, i \leq c$ are starting locations of some clients and all $v_{i'}, i' > c$ are not starting locations of some client. Before this local search step all clients have some facility at their start location and each facility has moved only one step. This means the local search operation that only moves $f_{i_1}, \ldots, f_{i_c}$ to locations $v_1, \ldots, v_c$ does no worse than the operation that moves all k facilities since we save at most one step for each $f_{i_j}, c < j \leq k$ and each client collocated with f_{i_j} must now move at least one step. It is also not hard to see that permuting the destinations of any $k \leq p$ facilities will not improve the cost so the current solution is a local minimum.

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 $F/(p + 2)$.

5 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 Δ 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 σC^*. To bring facilities to these k locations costs at most $k\Delta = \frac{\sigma}{\alpha n}$. Thus we have:

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

since $C^* \geq 1$.

Now suppose there is an α-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 [20], is all that is required for this result. Given a graph $G(V, E)$ and an integer k, the vertex cover problem is to determine if there is a collection of nodes $C \subseteq V$ with $|C| \leq k$ such that each edge E has one of its endpoints in C. From such a instance of the vertex cover problem, 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.

If G has a vertex cover of size k, then we can obtain a solution in H with maximum movement 1 by moving the k facilities to the vertices in V that correspond to some vertex cover of size k. Then each client $e \in E$ is adjacent to some vertex in V with a facility which can be reached with cost 1. Similarly, if there is a solution of with maximum movement of 1 in H, then it is easy to see that all clients must meet facilities at nodes in v which corresponds to a vertex cover of G of size at most k. This implies the cost of a solution in H is 2 if G does not have a vertex cover of size k (since we can always move all the clients to a vertex f_i with maximum movement cost being 2).

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. □

6 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.

Another direction is to consider a more general version of TM-MFL in which 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 does not work for this more general setting. For example, we cannot bound the change in the cost of the solution after performing Phase 2 of our rounding (relocation of facilities). Note that this is also a problem when trying to balance our approximation ratio. That is, our algorithm find a solution of cost at most $8\overline{C} + 4\overline{F}$. A standard trick is to scale the costs of the facilities by some constant to improve the overall approximation guarantee. However, the proof of lemma 2.4 requires the movement cost of each facility to be bound by the movement cost of each client (captured by $1 \leq D_i$ in our setting) so such scaling is not possible.

As mentioned in [11], 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 [11] and this paper.

Acknowledgements: The second author thanks M. Hajiaghayi for some initial discussions about the problem. We also thank anonymous referees for their comments and suggestions.

References

[1] K. Aardal, F. Chudak, and D.B. Shmoys, *A 3-approximation for the k-level uncapacitiated facility location problem*, Information Processing Letters, 72 (1999), pp. 161–167.

[2] A. Armon, *On min-max r-gatherings*, To appear in Theoretical Computer Science. Earlier version in proceedings of the 5th international conference on Approximation and online algorithms, Lecture Notes in Computer Science, Volume 4927, year 2008. pp. 128-141

[3] 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.

[4] 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.

[5] S. Bespamyatnikh, B. Bhattacharya, D. Kirkpatrick, and M. Segal, *Mobile Facility Location*, In Proc. of 4th intl. workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM), 2000, pp. 46–53.

[6] 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.

[7] Jaroslaw Byrka, *An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem*, In Proceedings of APPROX 2007, pp. 29–43.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] Z. Friggstad, *Minimizing Movement in Mobile Facility Location Problems* M.Sc. thesis, University of Alberta, 2007.

[14] 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.

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

[16] 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.

[17] 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.

[18] 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.

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

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

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

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

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

[24] 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.

[25] 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.

[26] P. Zhang, *A new approximation algorithm for the k-facility location problem*, Theor. Comp. Sci., 384(1) (2007) pp. 126–135.

[27] J. Zhang, *Approximating the two-level facility location problem via a quasi-greedy approach*, In Proc. of 15th annual ACM-SIAM Symp. on Discrete Algorithms (SODA), 2004, pp. 801–810.