Structural Iterative Rounding for Generalized k-Median Problems

Anupam Gupta ✉
Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA

Benjamin Moseley ✉
Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA

Rudy Zhou ✉
Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

This paper considers approximation algorithms for generalized k-median problems. This class of problems can be informally described as k-median with a constant number of extra constraints, and includes k-median with outliers, and knapsack median. Our first contribution is a pseudo-approximation algorithm for generalized k-median that outputs a 6.387-approximate solution with a constant number of fractional variables. The algorithm is based on iteratively rounding linear programs, and the main technical innovation comes from understanding the rich structure of the resulting extreme points.

Using our pseudo-approximation algorithm, we give improved approximation algorithms for k-median with outliers and knapsack median. This involves combining our pseudo-approximation with pre- and post-processing steps to round a constant number of fractional variables at a small increase in cost. Our algorithms achieve approximation ratios $6.994 + \epsilon$ and $6.387 + \epsilon$ for k-median with outliers and knapsack median, respectively. These both improve on the best known approximations.

2012 ACM Subject Classification Theory of computation → Facility location and clustering

Keywords and phrases approximation algorithms, clustering, linear programming

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.77

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2009.00808`

Funding *Anupam Gupta*: Supported in part by NSF awards CCF-1907820, CCF-1955785, and CCF-2006953.
Benjamin Moseley: Supported in part by a Google Research Award, an Infor Research Award, a Carnegie Bosch Junior Faculty Chair and NSF grants CCF-1824303, CCF-1845146, CCF-1733873 and CMMI-1938909.

1 Introduction

Clustering is a fundamental problem in combinatorial optimization, where we wish to partition a set of data points into *clusters* such that points within the same cluster are more similar than points across different clusters. In this paper, we focus on generalizations of the *k-median* problem. Recall that in this problem, we are given a set F of facilities, a set C of clients, a metric d on $F \cup C$, and a parameter $k \in \mathbb{N}$. The goal is to choose a set $S \subset F$ of k facilities to open to minimize the sum of *connection costs* of each client to its closest open facility. That is, to minimize the objective $\sum_{j \in C} d(j, S)$, where we define $d(j, S) = \min_{i \in S} d(i, j)$.

The k-median problem is well-studied from the perspective of approximation algorithms, and many new algorithmic techniques have been discovered while studying it. Examples include linear program rounding [3, 13], primal-dual algorithms [10], and local search [1].

48th International Colloquium on Automata, Languages, and Programming (ICALP 2021).
Editors: Nikhil Bansal, Emanuela Merelli, and James Worrell; Article No. 77; pp. 77:1–77:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Recently, there has been significant interest in generalizations of the k-median problem [4, 11]. One such generalization is the *knapsack median* problem. In knapsack median, each facility has a non-negative weight, and we are given budget $B \geq 0$. The goal is to choose a set of open facilities of total weight at most B (instead of having cardinality at most k) to minimize the same objective function. That is, the open facilities must satisfy a knapsack constraint. Another commonly-studied generalization is *k-median with outliers*, also known as *robust k-median*. Here we open k facilities S, as in basic k-median, but we no longer have to serve all clients; now, we are only required to serve at least m clients $C' \subset C$ of our choice. Formally, the objective function is now $\sum_{j \in C'} d(j, S)$.

Both knapsack median and k-median with outliers have proven to be much more difficult than the standard k-median problem. Algorithmic techniques that have been successful in approximating k-median often lead to only a pseudo-approximation for these generalizations – that is, they violate the knapsack constraint or serve fewer than m clients [2, 4, 6, 9]. Obtaining "true" approximation algorithms requires new ideas beyond those of k-median. Currently the best approximation ratio for both problems is $7.081 + \epsilon$ due to the beautiful iterative rounding framework of Krishnaswamy, Li, and Sandeep [12]. The first and only other true approximation for k-median with outliers is a local search algorithm due to Ke Chen [5].

Generalized k-Median

Both knapsack median and k-median with outliers maintain the salient features of k-median; that is, the goal is to open facilities to minimize the connection costs of served clients. These variants differ in the way we put constraints on the open facilities and served clients. For example, in k-median with outliers, we are constrained to open at most k facilities, and serve at least m clients.

In this paper, we consider a further generalization of k-median that we call *generalized k-median (GKM)*. As in k-median, our goal is to open facilities to minimize the connection costs of served clients. In GKM, the open facilities must satisfy r_1 given knapsack constraints, and the served clients must satisfy r_2 given coverage constraints. We define $r = r_1 + r_2$ to be the number of side constraints overall.

For each knapsack constraint, we have a unique non-negative budget and each facility has a non-negative cost with respect to that budget. The open facilities satisfy all budgets. Similarly, for each coverage constraint, we have a unique non-negative quota and each client has a non-negative value with respect to that quota. Then the served clients must satisfy all quotas.

1.1 Our Results

The main contribution of this paper is a refined iterative rounding algorithm for GKM. Specifically, we show how to round the natural linear program (LP) relaxation of GKM to ensure all except $O(r)$ of the variables are integral, and the objective function is increased by at most a 6.387-factor. It is not difficult to show that the iterative rounding framework in [12] can be extended to show a similar result. Indeed, a 7.081-approximation for GKM with at most $O(r)$ fractional facilities is implicit in their work. The improvement in this work is the smaller loss in the objective value.

▶ **Theorem 1** (Pseudo-Approximation for GKM). *There exists a poly-time pseudo-approximation for GKM that outputs a solution of cost at most $6.387 \cdot Opt$ with at most $O(r)$ fractional facilities.*

Our improvement relies on analyzing the extreme points of certain set-cover-like LPs. These extreme points arise at the intermediate steps of our iterative rounding, and by using their structural properties, we obtain our improved pseudo-approximation for GKM. This work reveals some of the structure of such extreme points, and it shows how this structure can lead to improvements.

Our second contribution is improved "true" approximation algorithms for two special cases of GKM: knapsack median and k-median with outliers. For both problems, applying the pseudo-approximation algorithm for GKM gives a solution with $O(1)$ fractional facilities. Thus, the remaining work is to round a constant number of fractional facilities to obtain an integral solution. To achieve this goal, we apply known sparsification techniques [12] to pre-process the instance, and then develop new post-processing algorithms to round the final $O(1)$ fractional facilities.

We show how to round these remaining variables for knapsack median at arbitrarily small loss, giving a $6.387+\epsilon$-approximation, improving on the best $7.081+\epsilon$-approximation. For k-median with outliers, a more sophisticated post-processing is needed to round the $O(1)$ fractional facilities. This procedure loses more in the approximation ratio. In the end, we obtain a $6.994+\epsilon$-approximation, modestly improving on the best known $7.081+\epsilon$-approximation.

▶ **Theorem 2** (Approximation for Knapsack Median)**.** *For any $\epsilon > 0$, there exists a $n^{(1/\epsilon)}$-time $(6.387+\epsilon)$-approximation for knapsack median.*

▶ **Theorem 3** (Approximation for k-Median with Outliers)**.** *For any $\epsilon > 0$, there exists a $n^{(1/\epsilon)}$-time $(6.994+\epsilon)$-approximation for k-median with outliers.*

Organization

In this paper, we develop and analyze the pseudo-approximation algorithm for GKM guaranteed by Theorem 1. We defer the "true" approximation algorithms guaranteed by Theorems 2 and 3 to the full version of this paper [8], §6.

1.2 Overview of Techniques

To illustrate our techniques, we first introduce a natural LP relaxations for GKM. The problem admits an integer program formulation, with variables $\{x_{ij}\}_{i\in F, j\in C}$ and $\{y_i\}_{i\in F}$, where x_{ij} indicates that *client j connects to facility i* and y_i indicates that *facility i is open.* Relaxing the integrality constraints gives the linear program relaxation LP_1. We focus on only LP_1 for now.

$$
(LP_1)\ \min_{x,y} \quad \begin{array}{ll} \sum_{i\in F}\sum_{j\in C} d(i,j)\,x_{ij} & \\ \sum_{i\in F} x_{ij} \le 1 & \forall j \in C \\ x_{ij} \le y_i & \forall i\in F, j\in C \\ Wy \le b & \\ \sum_{j\in C} a_j(\sum_{i\in F} x_{ij}) \ge c & \\ x_{ij}, y_i \in [0,1] & \forall i \in F, j\in C \end{array}
\qquad\qquad
(LP_2):\ \min_y \quad \begin{array}{ll} \sum_{j\in C}\sum_{i\in F_j} d(i,j)\,y_i & \\ y(F_j) \le 1 & \forall j\in C \\ & \\ Wy \le b & \\ \sum_{j\in C} a_j y(F_j) \ge c & \\ y_i \in [0,1] & \forall i \in F \end{array}
$$

The linear program LP_1 is the standard k-median LP with the extra side constraints. Note that $\sum_{i\in F} x_{ij} \le 1$ may seem opposite to the intuition that we want clients to get "enough" coverage from the facilities, but that will be guaranteed by the coverage constraints below.

The constraint $Wy \le b$ corresponds to the r_1 knapsack constraints on the facilities y, where $W \in \mathbb{R}_+^{r_1\times F}$ and $b \in \mathbb{R}_+^{r_1}$. These r_1 packing constraints can be thought of as a multidimensional knapsack constraint over the facilities, and ensure that "few" facilities

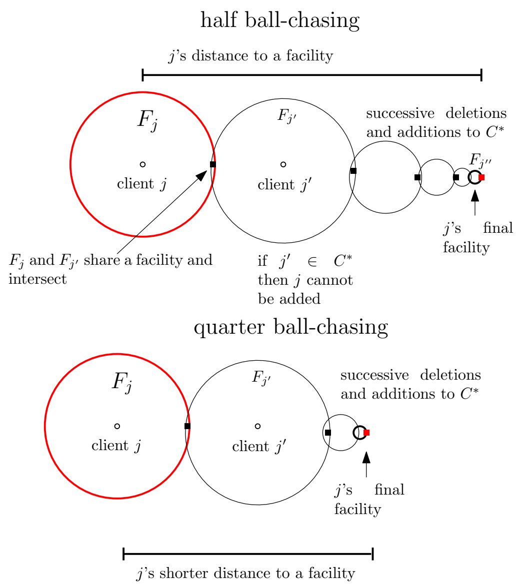

Figure 1 Half and quarter ball chasing.

are opened. Next, $\sum_{j\in C} a_j(\sum_i x_{ij}) \geq c$ corresponds to the r_2 coverage constraints on the clients, where $a_j \in \mathbb{R}_+^{r_2}$ for all $j \in C$ and $c \in \mathbb{R}_+^{r_2}$. These coverage constraints ensure that "enough" clients are served. E.g., having one packing constraint $\sum_{i\in F} y_i \leq k$ and one covering constraint $\sum_{j\in C}\sum_{i\in F} x_{ij} \geq m$ ensures that at least m clients are covered by at most k facilities; this is the k-median with outliers problem.

1.2.1 Constructing LP_2

The idea from [12] is to prescribe a set $F_j \subseteq F$ of permissible facilities for each client j such that x_{ij} is implicitly set to $y_i\mathbf{1}(i \in F_j)$. The procedure to construct these F_j's is given in Proposition 4. Using this procedure, LP_2 is also a relaxation for GKM. Note that in LP_2, we use the notation $y(F') = \sum_{i\in F'} y_i$ for $F' \subset F$.

Now consider solving LP_2 to obtain an optimal extreme point $\bar{y}$. There must be $|F|$ linearly independent tight constraints at $\bar{y}$. The tight constraints of interest are the $y(F_j) \leq 1$ constraints; in general, there are at most $|C|$ such tight constraints, and we have little structural understanding of the F_j-sets.

1.2.2 Prior Iterative Rounding Framework

Consider the family of F_j-sets corresponding to tight constraints, so $\mathcal{F} = \{F_j \mid j \in C,\ \bar{y}(F_j) = 1\}$. If $\mathcal{F}$ is a family of disjoint sets , then the tight constraints of LP_2 form a face of a partition matroid polytope intersected with at most r side constraints (the knapsack and coverage constraints.) Using ideas from, e.g., [12, 7], we can show that $\bar{y}$ has at most $O(r)$ fractional variables.

Indeed, the goal of the iterative rounding framework in [12] is to control the set family $\mathcal{F}$ to obtain an optimal extreme point where $\mathcal{F}$ is a disjoint family. To achieve this goal, they iteratively round an auxiliary LP based on LP_2, where they have the constraint $y(F_j) = 1$ for all clients j in a special set $C^* \subset C$. Roughly, they regulate what clients are added to C^* and delete constraints $y(F_j) \leq 1$ for some clients. The idea is that a client j whose constraint is deleted must be close to some client j' in C^*. Since $y(F_{j'}) = 1$ we can serve j with the facility for j'; the cost is small if j''s facility is close to j.

To get intuition, assume each client j can pay the farthest distance to a facility in F_j, and call this the *radius* of F_j. (Precisely, clients may not be able to afford this distance, but we use this assumption to highlight the ideas behind our algorithmic decisions.) For simplicity, assume every F_j-set is a ball whose radius is a power of two. Over time, this radius shrinks if some y-variables in F_j are set to zero. Consider applying the following iterative steps until none are applicable, in which case C^* corresponds to the tight constraints: (1) delete a constraint for $j \notin C^*$ if the radius of F_j is at least that of some $F_{j'}$ for $j' \in C^*$ and $F_j \cap F_{j'} \neq \emptyset$. (2) add $j \notin C^*$ to C^* if $y(F_j) \leq 1$ is tight and for every $j' \in C^*$ such that $F_j \cap F_{j'} \neq \emptyset$ it is the case that $F_{j'}$ has a radius strictly larger than F_j. If added then remove all j' from C^* where j's radius is *half* or less of the radius of j' and $F_j \cap F_{j'} \neq \emptyset$.

The approximation ratio is bounded by how much a client j with a deleted constraint pays to get to a facility serving a client in C^*. After removing j's constraint, the case to worry about is if j's closest client $j' \in C^*$ is later removed from C^*. This happens only if j'' is added to C^*, with $F_{j''}$ having half the radius of $F_{j'}$. Thus every time we remove j's closest client in C^*, we guarantee that j's cost only increases geometrically. The approximation ratio is proportional to the total distance that j must travel and can be directly related to the distance of "ball-chasing" though these F_j sets. When we remove a client j from C^* due to $j' \in C^*$ such that $F_{j'} \cap F_j \neq \emptyset$ and j' has radius at most half of j, we call this a *half-chasing step*. See Figure 1.

1.2.3 New Framework via Structured Extreme Points

The target of our framework is to ensure that the radius decreases in the ball-chasing at a faster rate, in particular *one-quarter*. This gives closer facilities for clients whose constraints are deleted. See Figure 1. To achieve this *quarter-chasing step*, we can simply change half to one-quarter in step (2) above.

Making this change immediately decreases the approximation ratio; however, the challenge is that $\mathcal{F}$ is no longer disjoint. Indeed, it can be the case that $j, j' \in C^*$ such that $F_j \cap F_{j'} \neq \emptyset$ if their radii differ by only a one half factor. Instead, our quarter ball-chasing algorithm maintains that $\mathcal{F}$ is not disjoint, but has a *bipartite intersection graph*.

The main technical challenge is obtaining an extreme point with $O(r)$ fractional variables, which is no longer guaranteed as when $\mathcal{F}$ was disjoint. Indeed, if $\mathcal{F}$ has bipartite intersection graph, then the tight constraints form a face of the intersection of two partition matroid polytopes intersected with at most r side constraints. In general, we *cannot upper bound the number of fractional variables* arising in the extreme points of such polytopes. However, such extreme points have a nice combinatorial structure: the intersection graph can be decomposed into $O(r)$ disjoint paths. We exploit this "chain decomposition" of extreme points arising in our iterative rounding to discover clients j that can be removed from C^* even if there is not a $j' \in C^*$ where $F_{j'}$ has one quarter of the radius of F_j. We continue this procedure until we are left with only $O(r)$ fractional variables.

The main technical contribution of this work is showing how the problem can be reduced to structural characterization of extreme points corresponding to bipartite matching. This illustrates some of the structural properties of polytopes defined by k-median-type problems. We hope that this helps lead to other structural characterizations of these polytopes and ultimately improved algorithms.

2 Auxiliary LP for Iterative Rounding

In this section, we construct the auxiliary LP, LP_{iter}. We note that we use the same relaxation used in [12]. Recall the two goals of iterative rounding, outlined in the technical overview; we want to maintain a set of clients $C^* \subset C$ such that $\{F_j \mid j \in C^*\}$ has bipartite intersection graph, and C^* should provide a good set of open facilities for the clients that are not in C^*. Thus, we want to define LP_{iter} to accommodate moving clients in and out of C^*, while having the LP faithfully capture how much we think the clients outside of C^* should pay in connection costs.

2.1 Defining F-balls

Our starting point is LP_2, so we assume that we have sets $F_j \subset F$ for all $j \in C$. The next proposition states that such sets can be found efficiently so that LP_2 is a relaxation of GKM.

▶ **Proposition 4.** *There is a poly-time algorithm that given GKM instance $\mathcal{I}$ outputs sets $F_j \subseteq F$ for $j \in C$ such that $Opt(LP_2) \leq Opt(\mathcal{I})$.*

Proof. Let $\mathcal{I}$ be the given instance of GKM and (x^*, y^*) be an optimal solution to LP_1.

Observe that if $x^*_{ij} \in \{0, y^*_i\}$ for all $i \in F, j \in C$, then we can define $F_j = \{i \in F \mid x^*_{ij} > 0\}$ for all $j \in C$. It is easy to verify in this case that y^* is feasible for LP_2 and achieves the same objective value in LP_2 as (x^*, y^*) achieves in LP_1, which completes the proof.

Thus our goal is to duplicate facilities in F and re-allocate the x- and y-values appropriately until $x^*_{ij} \in \{0, y^*_i\}$ for all $i \in F, j \in C$. To prevent confusion, let F denote the original set of facilities, and let F' denote the modified set of facilities, where make $n = |C|$ copies of each facility in F, so for each $i \in F$, we have copies $i_1, \dots, i_n \in F'$.

Now we define $x' \in [0,1]^{F' \times C}$ and $y' \in [0,1]^{F'}$ with the desired properties. For each $i \in F$, we assume without loss of generality that $0 \leq x_{i1} \leq x_{i2} \leq \dots \leq x_{in} \leq y_i$. We define $x'_{i_1 1}, \dots, x'_{i_n n}$ and $y'_{i_1}, \dots, y'_{i_n}$ recursively:

Let $y'_{i_1} = x_{i1}$ and $x'_{i_1 j} = x_{ij}$ for all $j \in [n]$.

Now for $k > 1$, let $y'_{i_k} = x_{ik} - x_{i(k-1)}$ and $x'_{i_k j} = \begin{cases} 0 & , j < k \\ y'_{i_k} & , j \geq k \end{cases}$ for all $j \in [n]$.

It is easy to verify that (x', y') is feasible for LP_1 (after duplicating facilities) and $x'_{ij} \in \{0, y'_i\}$ for all $i \in F', j \in C$, as required. Further, it is clear that this algorithm is polynomial time. ◀

In the technical overview, we assumed the radii of the F_j sets were powers of two. To formalize this idea, we discretize the distances to powers of $\tau > 1$ (up to some random offset.) The choice of τ is to optimize the final approximation ratio. The main ideas of the algorithm remain the same if we discretize to powers of, say 2, with no random offset. Our discretization procedure is the following:

Fix some $\tau > 1$ and sample the random offset $\alpha \in [1, \tau)$ such that $\log_e \alpha$ is uniformly distributed in $[0, \log_e \tau)$. Without loss of generality, we may assume that the smallest non-zero inter-point distance is 1. Then we define the possible discretized distances, $L(-2) = -1, L(-1) = 0, \dots, L(\ell) = \alpha\tau^\ell$ for all $\ell \in \mathbb{N}$. For each $p, q \in F \cup C$, we round $d(p,q)$ *up* to the next largest discretized distance. Let $d'(p,q)$ denote the rounded distances. Observe that $d(p,q) \leq d'(p,q)$ for all $p, q \in F \cup C$. The next proposition bounds the cost of discretization.

▶ **Proposition 5.** *For all $p, q \in F \cup C$, we have $\mathbb{E}[d'(p,q)] = \frac{\tau - 1}{\log_e \tau} d(p,q)$*

Proof. If $d(p,q) = 0$, then the claim is trivial. Suppose $d(p,q) \geq 1$. We can rewrite $d(p,q) = \tau^{\ell+f}$ for some $\ell \in \mathbb{N}, f \in [0,1)$. Also, for convenience we define $\beta = \log_\tau \alpha$. Because $\log_e \alpha$ is uniformly distributed in $[0, \log_e \tau)$, it follows that β is uniformly distributed in $[0,1)$.

It follows, $d(p,q)$ is rounded to $\alpha\tau^\ell = \tau^{\ell+\beta}$ exactly when $\beta \geq f$, and otherwise $d(p,q)$ is rounded to $\tau^{\ell+\beta+1}$ when $\beta < f$. Thus we compute:

$$
\begin{aligned}
\mathbb{E}[d'(p,q)] &= \int_{\beta=0}^{f} \tau^{\ell+\beta+1}\, d\beta + \int_{\beta=f}^{1} \tau^{\ell+\beta}\, d\beta \\
&= \frac{1}{\log_e \tau}(\tau^{\ell+\beta+1}|_{\beta=0}^{f} + \tau^{\ell+\beta}|_{\beta=f}^{1}) \\
&= \frac{1}{\log_e \tau}(\tau^{\ell+f+1} - \tau^{\ell+1} + \tau^{\ell+1} - \tau^{\ell+f}) \\
&= \frac{1}{\log_e \tau}(\tau^{\ell+f+1} - \tau^{\ell+f}) \\
&= \frac{\tau - 1}{\log_e \tau} d(p,q).
\end{aligned}
$$

◀

Now using the discretized distances, we can define the *radius level* of F_j for all $j \in C$ by:

$$\ell_j = \min_{\ell \geq -1}\{\ell \mid d'(j,i) \leq L(\ell) \quad \forall i \in F_j\}.$$

One should imagine that F_j is a ball of radius $L(\ell_j)$ in terms of the d'-distances. Thus, we will often refer to F_j as the *F-ball of client j*. Further, to accommodate "shrinking" the F_j sets, we define the *inner ball of F_j* by:

$$B_j = \{i \in F_j \mid d'(j,i) \leq L(\ell_j - 1)\}.$$

Note that we defined $L(-2) = -1$ so that if $\ell_j = -1$, then $B_j = \emptyset$.

2.2 Constructing LP_{iter}

Our auxiliary LP will maintain three sets of clients: C_{part}, C_{full}, and C^*. C_{part} consists of all clients, whom we have not yet decided whether we should serve them or not. Then for all clients in C_{full} and C^*, we decide to serve them fully. The difference between the clients in C_{full} and C^* is that for the former, we remove the constraint $y(F_j) = 1$ from the LP, while for the latter we still require $y(F_j) = 1$. Thus although we commit to serving C_{full}, such clients rely on C^* to find an open facility to connect to. Using the discretized distances, radius levels, inner balls, and these three sets of clients, we are ready to define LP_{iter}:

$$
\min_y \sum_{j \in C_{part}} \sum_{i \in F_j} d'(i,j) y_i + \sum_{j \in C_{full} \cup C^*} \Big(\sum_{i \in B_j} d'(i,j) y_i + (1 - y(B_j)) L(\ell_j)\Big) \qquad (LP_{iter})
$$

$$
\begin{aligned}
\text{s.t.}\quad & y(F_j) \leq 1 \quad \forall j \in C_{part} \\
& y(B_j) \leq 1 \quad \forall j \in C_{full} \\
& y(F_j) = 1 \quad \forall j \in C^* \\
& Wy \leq b \\
& \sum_{j \in C_{part}} a_j y(F_j) \geq c - \sum_{j \in C_{full} \cup C^*} a_j \\
& 0 \leq y \leq 1
\end{aligned}
$$

This completes the construction of LP_{iter}. Note that we use the *rounded* distances in the definition of LP_{iter} rather than the original distances. Keeping this in mind, if $C_{part} = C$ and $C_{full}, C^* = \emptyset$, then LP_{iter} is the same as LP_2 up to the discretized distances, so the following lemma is immediate. The algorithm described by the lemma is exactly the steps we took in this section.

▶ **Lemma 6.** *There exists a poly-time algorithm that takes as input a GKM instance $\mathcal{I}$ and outputs LP_{iter} such that $\mathbb{E}[Opt(LP_{iter})] \leq \frac{\tau-1}{\log_e \tau} Opt(\mathcal{I})$.*

The remainder of the paper shows how to iterative round LP_{iter} to obtain our pseudo-approximation for GKM.

2.3 Understanding LP_{iter}

Initially, all clients are in C_{part}. For clients in C_{part}, we are not sure yet whether we should serve them or not. Thus for these clients, we simply require $y(F_j) \leq 1$, so they can be served any amount, and in the objective, the contribution of a client from C_{part} is exactly its connection cost (up to discretization) to F_j.

The clients in C_{full} correspond to the "deleted" constraints in the technical overview. Importantly, for $j \in C_{full}$, we do not require that $y(F_j) = 1$; rather, we relax this condition to $y(B_j) \leq 1$. Recall that we made the assumption that every client can pay the radius of its F_j set. To realize this idea, we require that each $j \in C_{full}$ pays its connection costs to B_j in the objective. Then, to serve j fully, j must find $(1 - y(B_j))$ units of open facility to connect to beyond B_j. Now j truly pays its radius, $L(\ell_j)$, for this $(1 - y(B_j))$ units of connections in LP_{iter}, so we can do ball-chasing to C^* to find these facilities. In this case, we say that we *re-route* the client j to some *destination*.

Note that using the discretized distances, a half-chasing step corresponds to intersecting a neighboring ball of one radius level smaller, and a quarter-chasing step is analogously defined.

For clients in C^*, we require $y(F_j) = 1$. Note that the contribution of a $j \in C^*$ to the objective of LP_{iter} is exactly its connection cost to F_j. The purpose of C^* is to provide destinations for C_{full}.

Finally, because we have decided to fully serve all clients in C_{full} and C^*, regardless of how much they are actually served in their F-balls, we imagine that they every $j \in C_{full} \cup C^*$ contributes a_j to the coverage constraints, which is reflected in LP_{iter}.

3 Basic Iterative Rounding Phase

In this section, we describe the iterative rounding phase of our algorithm. This phase has two main goals: (a) to simplify the constraint set of LP_{iter}, and (b) to decide which clients to serve and how to serve them. To make these two decisions, we repeatedly solve LP_{iter} to obtain an optimal extreme point, and then use the structure of tight constraints to update LP_{iter}, and reroute clients accordingly.

Throughout our algorithm, we will modify the data of LP_{iter} - we will move clients between C_{part}, C_{full}, and C^* and modify the F-balls and radius levels. The key property that we wish to maintain is the *Distinct Neighbors Property*.

▶ **Definition 7** (Distinct Neighbors Property). *For all $j_1, j_2 \in C^*$, if $F_{j_1} \cap F_{j_2} \neq \emptyset$, then $|\ell_{j_1} - \ell_{j_2}| = 1$. In words, if the F-balls of two clients in C^* intersect, then they differ by exactly one radius level.*

This simple property will enable quarter-chasing and a structural characterization of the extreme points of LP_{iter} - both of which are crucial to our improved algorithm.

3.1 The Algorithm

Our algorithm repeatedly solves LP_{iter} to obtain an optimal extreme point $\bar{y}$, and then performs one of the following three possible updates, based on the tight constraints:

1. If some facility i is set to zero in $\bar{y}$, we delete it from the instance.
2. If constraint $\bar{y}(F_j) \leq 1$ is tight for some $j \in C_{part}$, then we decide to fully serve client j by moving j to either C_{full} or C^*. Initially, we add j to C_{full} then run Algorithm 2 to decide if j should be in C^* instead.
3. If constraint $\bar{y}(B_j) \leq 1$ is tight for some $j \in C_{full}$, we shrink F_j by one radius level (so j's new F-ball is exactly B_j.) Then we possibly move j to C^* by running Algorithm 2 for j.

These steps are made formal in Algorithms 1 (IterativeRound) and 2 (ReRoute). IterativeRound relies on the subroutine ReRoute, which gives our criterion for moving a client to C^*. This criterion for adding clients to C^* is the key way in which our algorithm differs from that of [12]. In [12], the criterion used ensures that $\{F_j \mid j \in C^*\}$ is a family of disjoint sets. In contrast, we allow F-balls for clients in C^* to intersect, as long as they satisfy the Distinct Neighbors Property. Thus, our algorithm allows for rich structures in the set system $\{F_j \mid j \in C^*\}$.

Algorithm 1 IterativeRound.

Input: LP_{iter}
Result: Modifies LP_{iter} and outputs an optimal extreme point of LP_{iter}

repeat
  Solve LP_{iter} to obtain optimal extreme point $\bar{y}$.
  if *there exists a facility* $i \in F$ *such that* $\bar{y}_i \geq 0$ *is tight* **then**
    Delete i from F.
  else if *there exists a client* $j \in C_{part}$ *such that* $y(F_j) \leq 1$ *is tight* **then**
    Move j from C_{part} to C_{full}.
    ReRoute(j)
  else if *there exists a client* $j \in C_{full}$ *such that* $\bar{y}(B_j) \leq 1$ *is tight* **then**
    Update $F_j \leftarrow B_j$ and decrement ℓ_j by 1.
    Update $B_j \leftarrow \{i \in F_j \mid d'(j,i) \leq L(\ell_j - 1)\}$.
    ReRoute(j)
  else
    Output $\bar{y}$ and Terminate.
until *termination*

The modifications made by IterativeRound do not increase $Opt(LP_{iter})$, so upon termination of our algorithm, we have an optimal extreme point $\bar{y}$ to LP_{iter} such that LP_{iter} is still a relaxation of GKM and no non-negativity constraint, C_{part}-constraint, or C_{full}-constraint is tight for $\bar{y}$. Further, it is easy to check that the Distinct Neighbors Property is maintained.

Algorithm 2 REROUTE.

Input: Client $j \in C_{full}$

Result: Decide whether to move j to C^* or not

if $\ell_j \leq \ell_{j'} - 1$ *for all* $j' \in C^*$ *such that* $F_j \cap F_{j'} \neq \emptyset$ **then**

  Move j from C_{full} to C^*.

  For all $j' \in C^*$ such that $F_j \cap F_{j'} \neq \emptyset$ and $\ell_{j'} \geq \ell_j + 2$, move j' from C^* to C_{full}.

3.2 Sketch of Analysis

Recall the goals from the beginning of the section: procedure ITERATIVEROUND achieves goal (a) of making $\{F_j \mid j \in C^*\}$ simpler while maintaining the Distinct Neighbors Property. Since we moved facilities between C^* and C_{full}, achieving goal (b) means deciding which facilities to open, and guaranteeing that each client has a "close-by" open facility. (Recall from §2 that C^* is the set of clients such that their F_j-balls are guaranteed to contain an open facility, and C_{full} are the clients which are guaranteed to be served but using facilities opened in C^*.)

To achieve goal (b), we observe that REROUTE always gives quarter-chasing steps. That is, if we move a client j from C^* to C_{full}, then we are guaranteed a neighboring client $j' \in C^*$ with radius level at least two smaller than j. Thus, each time we re-route j to a further destination (i.e. if j' is subject to another quarter-chasing step), the extra distance j must travel decreases geometrically. In the end, we can show that j will have an open facility within $O(1)$ times its radius.

4 Iterative Operation for Structured Extreme Points

In this section, we achieve two goals: (a) we show that the structure of the extreme points of LP_{iter} obtained from ITERATIVEROUND are highly structured, and admit a *chain decomposition*. Then, (b) we exploit this chain decomposition to define a *new* iterative operation that is applicable whenever $\bar{y}$ has "many" (i.e., more than $O(r)$) fractional variables. We emphasize that this characterization of the extreme points is what enables the new iterative rounding algorithm.

4.1 Chain Decomposition

A chain is a sequence of clients in C^* where the F-ball of each client j contains exactly two facilities – one shared with the previous ball and other with the next.

▶ **Definition 8** (Chain). *A chain is a sequence of clients* $(j_1, \ldots, j_p) \subseteq C^*$ *satisfying:*

- $|F_{j_q}| = 2$ *for all* $q \in [p]$*, and*
- $F_{j_q} \cap F_{j_{q+1}} \neq \emptyset$ *for all* $q \in [p-1]$.

Our chain decomposition is a partition of the *fractional* C^*-clients given in the next theorem, which is our main structural characterization of the extreme points of LP_{iter}. (We say a client j is fractional if all facilities in F_j are fractional; we denote the fractional clients in C^* by $C^*_{<1}$.) We defer the proof of the next structural theorem to the full version of this paper [8] §8, and instead focus on how to apply it.

▶ **Theorem 9** (Chain Decomposition). *Upon termination of* ITERATIVEROUND*, there exists a partition of* $C^*_{<1}$ *into at most* $3r$ *chains, along with a set of at most* $2r$ *violating clients (clients that are not in any chain.)*

The proof relies on analyzing the extreme points of LP_{iter} satisfying the Distinct Neighbors Property. We show that this boils down to analyzing a bipartite matching polytope with r side constraints.

4.2 Iterative Operation for Chain Decompositions

Leveraging Theorem 9, consider an optimal extreme point $\bar{y}$ of LP_{iter}, and its chain decomposition. We show that if the number of fractional variables in $\bar{y}$ is sufficiently large, there exists a useful structure in the chain decomposition, which we call a *candidate configuration.*

▶ **Definition 10** (Candidate Config)**.** *Let $\bar{y}$ be an optimal extreme point of LP_{iter}. A candidate configuration is a pair of two clients $(j, j') \subset C^*_{<1}$ such that:*

1. $F_j \cap F_{j'} \neq \emptyset$
2. $\ell_{j'} \leq \ell_j - 1$
3. *Every facility in F_j and $F_{j'}$ is in at exactly two F-balls for clients in C^**
4. $|F_j| = 2$ *and* $|F_{j'}| = 2$

One should imagine that a candidate configuration is two neighboring balls on a sufficiently long chain.

▶ **Lemma 11.** *If* IterativeRound *outputs an extreme point that has at least $15r$ fractional facilities, then there exist a candidate configuration in $C^*_{<1}$.*

To prove Lemma 11, which bounds the number of fractional facilities needed to have a candidate configuration, we first prove a bound on the number of factional clients needed. The bound on the number of facilities will follow by a dimension argument.

▶ **Proposition 12.** *Suppose LP_{iter} satisfies the Distinct Neighbors Property. Then each facility is in at most two F-balls for clients in C^*.*

Proof. Assume for contradiction that there exists a facility i such that $i \in F_{j_1} \cap F_{j_2} \cap F_{j_3}$ for distinct clients $j_1, j_2, j_3 \in C^*$. Then j_1 and j_2 differ by one radius level, and j_2 and j_3 differ by one radius level. However, now it cannot be the case that j_1 and j_3 also differ by one radius level. This contradicts the Distinct Neighbors Property. ◀

▶ **Lemma 13.** *Suppose LP_{iter} satisfies all Basic Invariants, and let $\bar{y}$ be an optimal extreme point of LP_{iter} such that no C_{part}-, C_{full}-, or non-negativity constraint is tight. If $|C^*_{<1}| \geq 14r$, then there exist a candidate configuration in $C^*_{<1}$.*

Proof. We claim that in order for $C^*_{<1}$ to have a candidate configuration, it suffices to have a chain of length at least four in $C^*_{<1}$. To see this, let $(j_1, j_2, j_3, j_4, \dots) \subset C^*_{<1}$ be a chain of length at least four. Then $F_{j_2} \cap F_{j_3} \neq \emptyset$, and by the Distinct Neighbors Property, either $\ell_{j_3} = \ell_{j_2} - 1$ or $\ell_{j_2} = \ell_{j_3} - 1$.

We only consider the former case, because both cases are analogous. Thus, if $\ell_{j_3} = \ell_{j_2} - 1$, then we claim that (j_2, j_3) forms a candidate configuration. We already have the first two properties of a candidate configuration. Now we verify the last two. Because j_2 and j_3 are part of a chain, we have $|F_{j_2}| = 2$ and $|F_{j_3}| = 2$. Further, j_2 has neighbors j_1 and j_3 along the chain. By Proposition 12, each facility in F_{j_2} is in at most two F-balls for clients in C^*. In particular, one of the facilities in F_{j_2} is shared by F_{j_1} and F_{j_2}, and the other must be shared by F_{j_2} and F_{j_3}. Thus, each facility in F_{j_2} is in exactly two F-balls for clients in C^*. An analogous argument holds for F_{j_3}, so (j_2, j_3) satisfies all properties of a candidate configuration, as required.

Now suppose $|C^*_{<1}| \geq 14r$. By Theorem 9, $C^*_{<1}$ admits a chain decomposition into at most $3r$ chains and a set of at most $2r$ violating clients. Then at least $12r$ of the clients in $C^*_{<1}$ belong to the $3r$ chains. By averaging, there must exist a chain with size at least $\frac{12r}{3r} = 4$, as required. ◀

Now we relate the number of fractional facilities with the number of fractional C^*-clients by a dimension argument.

▶ **Lemma 14.** *Let $\bar{y}$ be an extreme point of LP_{iter} such that no C_{part}-, C_{full}-, or non-negativity constraint is tight. Then the number of fractional facilities in $\bar{y}$ satisfies $|F_{<1}| \leq |C^*_{<1}| + r$ (recall that r is the number of side constraints.)*

Proof of Lemma 14. We construct a basis $\bar{y}$. First, for each integral facility $i \in F_{=1}$, we add the integrality constraint $\bar{y}_i \leq 1$ to our basis. Thus we currently have $|F_{=1}|$ constraints in our basis.

It remains to choose $|F_{<1}|$ further linearly independent constraints to add to our basis. Note that we have already added all tight integrality constraints to our basis, and no non-negativity constraint is tight. Then the only remaining tight constraints we can add are the C^*-constraints and the r side constraint.

We claim that we cannot add any $C^*_{=1}$-constraints, because every $C^*_{=1}$-constraint is of the form $y(F_j) = y_{i_j} = 1$ for the unique integral facility $i_j \in F_1$. Note that here we used the fact that there is no facility that is set to zero. Thus every $C^*_{=1}$-constraint is linearly dependent with the tight integrality constraints, which we already chose.

It follows, the only possible constraints we can choose are the $C^*_{<1}$-constraints and the r side constraints so:

$$|F_{<1}| \leq |C^*_{<1}| + r.$$ ◀

Lemma 11 is immediate by composing the above two lemmas.

Proof of Lemma 11. By Lemma 13, it suffices to show that $|F_{<1}| \geq 15r$ implies that $|C^*_{<1}| \geq 14r$. Applying Lemma 14, we have:

$$15r \leq |F_{<1}| \leq |C^*_{<1}| + r.$$ ◀

Our new iterative operation is easy to state. Find a candidate configuration (j, j') and move j from C^* to C_{full}.

Algorithm 3 ConfigReRoute.

Input: An optimal extreme point $\bar{y}$ to LP_{iter} s.t. there exists a candidate configuration
Result: Modify LP_{iter}
1 Let $(j, j') \subset C^*_{<1}$ be any candidate configuration.
2 Move j from C^* to C_{full}.

It is easy to check that ConfigReRoute maintains the Distinct Neighbors Property and weakly decreases $Opt(LP_{iter})$.

4.3 Sketch of Analysis

The first two properties of candidate configurations are used to re-route j to j'. Observe a key difference between ReRoute and ConfigReRoute: In the former, we always guarantee quarter-chasing steps. On the other hand, in ConfigReRoute, we only guarantee a neighboring client of at least one radius level smaller, which corresponds to a half-chasing step. This raises the worry that if all re-routings are due to ConfigReRoute, any potential gains by ReRoute are not realized in the worst case. However we show that, roughly speaking, the last two properties of candidate configurations guarantee that the half-chasing steps of ConfigReRoute happen at most half the time.

In particular, suppose client j is re-routed via ConfigReRoute to j' , which is exactly one radius level smaller. If j' is later re-routed via ReRoute, then we can re-route j to j' and then this new destination. This gives one half- and one quarter-chasing step. The concern is if j' is later re-routed via ConfigReRoute, which would give j two half-chasing steps in a row. By analyzing the interactions between ReRoute and ConfigReRoute, we show that there must exist a j'' that gives j a quarter-chasing step. See Figure 2.

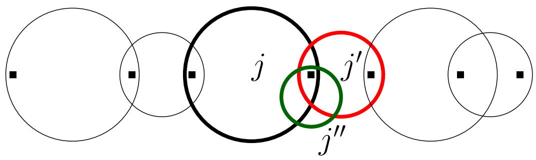

Figure 2 A chain of balls in C^*, where squares indicate facilities. First j is removed from C^* as part of candidate configuration (j, j'), so j' has strictly smaller radius than j. Then j'' is added to C^*, which has strictly smaller radius than j'. This gives j a destination that is at least two radius levels smaller.

5 Pseudo-Approximation Algorithm for GKM

The pseudo-approximation algorithm for GKM combine the iterative rounding algorithm IterativeRound from §3 with the re-routing operation ConfigReRoute from §4 to construct a solution to LP_{iter}. PseudoApproximation is the algorithm guaranteed by Theorem 1.

Algorithm 4 PseudoApproximation.

Input: LP_{iter}
Result: Modifies LP_{iter} and outputs an optimal extreme point of LP_{iter}

```
repeat
    Run IterativeRound to obtain an optimal extreme point ȳ of LP_iter
    if there exists a candidate configuration then
        Run ConfigReRoute
    else
        Output ȳ and Terminate
until Termination
```

5.1 Sketch of Analysis

There are two main components to analyzing PseudoApproximation. First, we show that the output extreme point has $O(r)$ fractional variables, which follows from Lemma 11. Second, we bound the re-routing cost, which follows from the sketches in §3 and §4. In particular, for each client, we can charge each of its half-chasing steps to a quarter-chasing step. This improves on [12], where every re-routing is via half-chasing steps. Optimizing the choice of τ (the discretization factor) gives our final approximation ratio.

5.2 Analysis of PseudoApproximation

In this section, we prove that PseudoApproximation satisfies the guarantees of Theorem 1. We begin by analyzing the runtime and number of fractional facilities.

▶ **Lemma 15.** *PseudoApproximation is a polynomial time algorithm that maintains the Distinct Neighbors Property, weakly decreases $Opt(LP_{iter})$, and outputs an optimal extreme point of LP_{iter} with at most $15r$ fractional variables.*

Proof of Lemma 15. We first show that both IterativeRound and ReRoute are polynomial time. It is clear that the latter runs in polynomial time. For IterativeRound, it suffices to show that the number of iterations of IterativeRound is polynomial. In each iteration, we make one of three actions. We either delete a facility from F, move a client from C_{part} to C_{full} or shrink a F-ball by one radius level for a client in $j \in C_{full}$.

We can delete each facility from F at most once, so we make at most $|F|$ deletions. Each client can move from C_{part} to C_{full} at most once, because we never move clients back from C_{full} to C_{part}, so we do this operations at most $|C|$ times. Finally, observe that $\ell_j \geq -1$ for all $j \in C$ over all iterations. We conclude that we can shrink each F-ball only polynomially many times.

For the runtime of PseudoApproximation, it suffices to show that the number of calls to IterativeRound and ConfigReRoute is polynomial.

In every iteration of PseudoApproximation, either we terminate or we are guaranteed to move a client from C^* to C_{full} in ConfigReRoute. Each client can be removed from C^* only polynomially many times, because each time a client is removed, in order to be re-added to C^*, it must be the case that we shrunk the F-ball of that client. However, again because $\ell_j \geq -1$ for all $j \in C$, we can shrink each F-ball only polynomially many times.

It is easy to check that both IterativeRound and ReRoute maintain the Distinct Neighbors Property and weakly decrease $Opt(LP_{iter})$.

Finally, upon termination of PseudoApproximation, there is no candidate configuration, so Lemma 11 implies that $\bar{y}$ has at most $15r$ fractional variables. ◀

5.3 Analysis of Re-Routing Cost

We now bound the re-routing cost by analyzing how C^* evolves throughout PseudoApproximation. This is one of the main technical contributions of our paper, and it is where our richer C^*-set and relaxed re-routing rules are used. [12] prove an analogous result about the re-routing cost of their algorithm. In the language of the following theorem statement, they show that $\alpha = \frac{\tau+1}{\tau-1}$ for the case $\beta = 1$. We improve on this factor by analyzing the interactions between ReRoute and ConfigReRoute. Interestingly, analyzing each of ReRoute and ConfigReRoute separately would not yield any improvement over [12] in the worst case, even with our richer set C^*. It is only by using the properties of candidate configurations and analyzing sequences of calls to ReRoute and ConfigReRoute that we get an improvement.

▶ **Theorem 16** (Re-Routing Cost). *Upon termination of* PseudoApproximation*, let* $S \subset F$ *be a set of open facilities and* $\beta \geq 1$ *such that* $d(j,S) \leq \beta L(\ell_j)$ *for all* $j \in C^*$*. Then for all* $j \in C_{full} \cup C^*$*,* $d(j,S) \leq (2+\alpha)L(\ell_j)$*, where* $\alpha = \max(\beta, 1 + \frac{1+\beta}{\tau}, \frac{\tau^3+2\tau^2+1}{\tau^3-1})$*.*

We will need the following discretized version of the triangle inequality.

▶ **Proposition 17.** *Let* $j, j' \in C$ *such that* F_j *and* $F_{j'}$ *intersect. Then* $d(j,j') \leq L(\ell_j) + L(\ell_{j'})$*.*

Proof. Let $i \in F_j \cap F_{j'}$. Then using the triangle inequality we can bound:

$$d(j,j') \leq d(j,i) + d(i,j') \leq d'(j,i) + d'(i,j') \leq L(\ell_j) + L(\ell_{j'}).$$ ◀

The next lemma analyzes the life-cycle of a client that enters C^* at some point in PseudoApproximation. Our improvement over [12] comes from this lemma.

▶ **Lemma 18.** *Upon termination of* PseudoApproximation*, let* $S \subset F$ *be a set of open facilities and* $\beta \geq 1$ *such that* $d(j,S) \leq \beta L(\ell_j)$ *for all* $j \in C^*$*. Suppose client* j *is added to* C^* *at radius level* ℓ *during* PseudoApproximation *(it may be removed later.) Then upon termination of* PseudoApproximation*, we have* $d(j,S) \leq \alpha L(\ell)$*, where* $\alpha = \max(\beta, 1 + \frac{1+\beta}{\tau}, \frac{\tau^3+2\tau^2+1}{\tau^3-1})$*.*

Proof. Consider a client j added to C^* with radius level ℓ. If j remains in C^* until termination, the lemma holds for j because $\alpha \geq \beta$. Thus, consider the case where j is later removed from C^* in PseudoApproximation. Note that the only two operations that can possibly cause this removal are ReRoute and ConfigReRoute. We prove the lemma by induction on $\ell = -1, 0, \ldots$. If $\ell = -1$, then j remains in C^* until termination because it has the smallest possible radius level and both ReRoute and ConfigReRoute remove a client from C^* only if there exists another client with strictly smaller radius level.

Similarly, if $\ell = 0$, we note that ReRoute removes a client from C^* only if there exists another client with radius level at least two smaller, which is not possible for j. Thus, if j does not remain in C^* until termination, there must exist some j' that is later added to C^* with radius level at most $\ell - 1 = -1$ such that $F_j \cap F_{j'} \neq \emptyset$. We know that j' remains in C^* until termination since it is of the lowest radius level. Thus:

$$d(j,S) \leq d(j,j') + d(j',S) \leq L(0) + L(-1) + \beta L(-1) = L(0).$$

Now consider $\ell > 0$ where j can possibly be removed from C^* by either ReRoute or ConfigReRoute. In the first case, j is removed by ReRoute, so there exists j' that is added to C^* such that $\ell_{j'} \leq \ell - 2$ and $F_j \cap F_{j'} \neq \emptyset$. Applying the inductive hypothesis to j', we can bound:

$$d(j,S) \leq d(j,j') + d(j',S) \leq L(\ell) + L(\ell-2) + \alpha L(\ell-2) \leq (1 + \frac{1+\alpha}{\tau^2})L(\ell).$$

It is easy to verify by routine calculations that $1 + \frac{1+\alpha}{\tau^2} \leq \alpha$ given that $\alpha \geq \frac{\tau^3+2\tau^2+1}{\tau^3-1}$.

For our final case, suppose j is removed by ConfigReRoute. Then there exists $j' \in C^*$ such that $F_j \cap F_{j'} \neq \emptyset$ and $\ell_{j'} \leq \ell - 1$. Further, $|F_{j'}| = 2$. If j' remains in C^* until termination, then:

$$d(j,S) \leq d(j,j') \leq L(\ell) + L(\ell-1) + \beta L(\ell-1) \leq (1 + \frac{1+\beta}{\tau})L(\ell).$$

Otherwise, j' is removed by ReRoute at an even later time because some j'' is added to C^* such that $\ell_{j''} \leq \ell_{j'} - 2$ and $F_{j'} \cap F_{j''} \neq \emptyset$. Applying the inductive hypothesis to j'', we can bound:

$$d(j,S) \leq d(j,j') + d(j',j'') + d(j'',S) \leq (1 + \frac{2}{\tau} + \frac{1+\alpha}{\tau^3})L(\ell).$$

where $\alpha \geq \frac{\tau^3+2\tau^2+1}{\tau^3-1}$ implies $1 + \frac{2}{\tau} + \frac{1+\alpha}{\tau^3} \leq \alpha$.

Now, we consider the case where j' is later removed by ConfigReRoute. To analyze this case, consider when j was removed by ConfigReRoute. At this time, we have $|F_{j'}| = 2$ by definition of Candidate Configuration. Because $F_j \cap F_{j'} \neq \emptyset$, consider any facility $i \in F_j \cap F_{j'}$. When j is removed from C^* by ConfigReRoute, we have that i is in exactly two F-balls for clients in C^*, exactly F_j and $F_{j'}$. However, after removing j from C^*, i is only in one F-ball for clients in C^* - namely $F_{j'}$.

Later, at the time j' is removed by ConfigReRoute, it must be the case that $|F_{j'}| = 2$ still, so $F_{j'}$ is unchanged between the time that j is removed and the time that j' is removed. Thus the facility i that was previously in $F_j \cap F_{j'}$ must still be present in $F_{j'}$. Then this facility must be in exactly two F-balls for clients in C^*, one of which is j'. It must be the case that the other F-ball containing i, say $F_{j''}$, was added to C^* between the removal of j and j'.

Note that the only operation that adds clients to C^* is ReRoute, so we consider the time between the removal of j and j' when j'' is added to C^*. Refer to Figure 2. At this time, we have $j' \in C^*$, and $F_{j'} \cap F_{j''} \neq \emptyset$ because of the facility i. Then it must be the case that j'' has strictly smaller radius level than j', so $\ell_{j''} \leq \ell_{j'} - 1 \leq \ell - 2$. To conclude the proof, we note that $F_j \cap F_{j''} \neq \emptyset$ due to the facility i, and apply the inductive hypothesis to j'':

$$d(j, S) \leq d(j, j'') + d(j'', S) \leq (1 + \frac{1+\alpha}{\tau^2})L(\ell,)$$

which is at most $\alpha L(\ell)$. ◀

Now using the above lemma, we can prove Theorem 16.

Proof of Theorem 16. Consider any client j that is in $C_{full} \cup C^*$ upon termination of PseudoApproximation. It must be the case that ReRoute(j) was called at least once during PseudoApproximation. Consider the time of the last such call to ReRoute(j). If j is added to C^* at this time, note that its radius level from now until termination remains unchanged, so applying Lemma 18 gives that $d(j, S) \leq \alpha L(\ell_j)$, as required. Otherwise, if j is not added to C^* at this time, then there must exist some $j' \in C^*$ such that $F_j \cap F_{j'} \neq \emptyset$ and $\ell_{j'} \leq \ell_j$. Then applying Lemma 18 to j', we have:

$$d(j, S) \leq d(j, j') + d(j', S) \leq L(\ell_j) + L(\ell_{j'}) + \alpha L(\ell_{j'}) \leq (2 + \alpha)L(\ell_j).$$ ◀

5.4 Putting it all Together: Pseudo-Approximation for GKM

In this section, we prove Theorem 1. In particular, we use the output of PseudoApproximation to construct a setting of the x-variables with the desired properties.

Proof of Theorem 1. Given as input an instance $\mathcal{I}$ of GKM, our algorithm is first to run the algorithm guaranteed by Lemma 6 to construct LP_{iter} from LP_1 such that $\mathbb{E}[Opt(LP_{iter})] \leq \frac{\tau-1}{\log_e \tau} Opt(\mathcal{I})$. Note that we will choose $\tau > 1$ later to optimize our final approximation ratio. Then we run PseudoApproximation on LP_{iter}, so by Theorem 15, PseudoApproximation outputs in polynomial time LP_{iter} along with an optimal solution $\bar{y}$ with $O(r)$ fractional variables.

Given $\bar{y}$, we define a setting $\bar{x}$ for the x-variables: for all $j \in C_{part}$, connect j to all facilities in F_j by setting $\bar{x}_{ij} = \bar{y}_i$ for all $i \in F_j$. For all $j \in C^*$, we have $\bar{y}(F_j) = 1$, so connect j to all facilities in F_j. Finally, to connect every $j \in C_{full}$ to one unit of open facilities, we use the following modification of Theorem 16:

▶ **Proposition 19.** *When* PseudoApproximation *terminates, for all $j \in C_{full} \cup C^*$, there exists one unit of open facilities with respect to $\bar{y}$ within distance $(2+\alpha)L(\ell_j)$ of j, where $\alpha = \max(1, 1+\frac{2}{\tau}, \frac{\tau^3+2\tau^2+1}{\tau^3-1})$.*

The proof of the above proposition is analogous to that of Theorem 16 in the case $\beta = 1$, so we omit it. To see this, note that for all $j \in C^*$, we have $\bar{y}(F_j) = 1$. This implies that each $j \in C^*$ has one unit of fractional facility within distance $L(\ell_j)$. Following an analogous inductive argument as in Lemma 18 gives the desired result.

By routine calculations, it is easy to see that $\alpha = \frac{\tau^3+2\tau^2+1}{\tau^3-1}$ for all $\tau > 1$. Now, for all $j \in C_{full}$, we connect j to all facilities in B_j. We want to connect j to one unit of open facilities, so to find the remaining $1 - \bar{y}(B_j)$ units, we connect j to an arbitrary $1 - \bar{y}(B_j)$ units of open facilities within distance $(2+\alpha)L(\ell_j)$ of j, whose existence is guaranteed by Proposition 19. This completes the description of $\bar{x}$.

It is easy to verify that $(\bar{x}, \bar{y})$ is feasible for LP_1, because $\bar{y}$ satisfies all knapsack constraints, and every client's contribution to the coverage constraints in LP_1 is exactly its contribution in LP_{iter}. Thus it remains to bound the cost of this solution. We claim that $LP_1(\bar{x}, \bar{y}) \leq (2+\alpha)Opt(LP_{iter})$, because each client in C_{part} and C^* contributes the same amount to LP_1 and LP_{iter} (up to discretization), and each client $j \in C_{full}$ has connection cost at most $2+\alpha$ times its contribution to LP_{iter}.

In conclusion, the expect cost of the solution $(\bar{x}, \bar{y})$ to LP_1 is at most:

$$(2+\alpha)\,\mathbb{E}[Opt(LP_{iter})] \leq \frac{\tau-1}{\log_e \tau}\left(2 + \frac{\tau^3+2\tau^2+1}{\tau^3-1}\right) Opt(\mathcal{I}).$$

Choosing $\tau > 1$ to minimize $\frac{\tau-1}{\log_e \tau}(2+\frac{\tau^3+2\tau^2+1}{\tau^3-1})$ gives $\tau = 2.046$ and $\frac{\tau-1}{\log_e \tau}(2+\frac{\tau^3+2\tau^2+1}{\tau^3-1}) = 6.387$. ◀

5.5 From Pseudo-Approximation to True Approximation

To extend PseudoApproximation to a true approximation algorithm for the special cases of knapsack median and k-median with outliers, we need to round the final $O(1)$ fractional facilities from the output of PseudoApproximation. To do so, we wrap PseudoApproximation with pre-processing and post-processing algorithms. The pre-processing involves enumeration to overcome the unbounded integrality gap, and the post-processing rounds the final $O(1)$ fractional facilities. See [8], §6 for details.

References

1 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004. doi:10.1137/S0097539702416402.

2 Jaroslaw Byrka, Thomas W. Pensyl, Bartosz Rybicki, Joachim Spoerhase, Aravind Srinivasan, and Khoa Trinh. An improved approximation algorithm for knapsack median using sparsification. *Algorithmica*, 80(4):1093–1114, 2018. doi:10.1007/s00453-017-0294-4.

3 Jaroslaw Byrka, Thomas W. Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k-median and positive correlation in budgeted optimization. *ACM Trans. Algorithms*, 13(2):23:1–23:31, 2017. doi:10.1145/2981561.

4 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA*, pages 642–651. ACM/SIAM, 2001.

5 Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In *SODA*, pages 826–835, 2008.

6 Zachary Friggstad, Kamyar Khodamoradi, Mohsen Rezapour, and Mohammad R. Salavatipour. Approximation schemes for clustering with outliers. *ACM Trans. Algorithms*, 15(2):26:1–26:26, 2019. doi:10.1145/3301446.

7 Fabrizio Grandoni, R. Ravi, Mohit Singh, and Rico Zenklusen. New approaches to multi-objective optimization. *Math. Program.*, 146(1-2):525–554, 2014. doi:10.1007/s10107-013-0703-7.

8 Anupam Gupta, Benjamin Moseley, and Rudy Zhou. Structural Iterative Rounding for Generalized k-Median Problems. *arXiv e-prints*, page arXiv:2009.00808, 2020. arXiv:2009.00808.

9 Sungjin Im, Mahshid Montazer Qaem, Benjamin Moseley, Xiaorui Sun, and Rudy Zhou. Fast noise removal for k-means clustering. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, pages 456–466, 2020.

10 Kamal Jain and Vijay V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *J. ACM*, 48(2):274–296, 2001. doi:10.1145/375827.375845.

11 Ravishankar Krishnaswamy, Amit Kumar, Viswanath Nagarajan, Yogish Sabharwal, and Barna Saha. Facility location with matroid or knapsack constraints. *Math. Oper. Res.*, 40(2):446–459, 2015. doi:10.1287/moor.2014.0678.

12 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 646–659. ACM, 2018. doi:10.1145/3188745.3188882.

13 Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. *SIAM J. Comput.*, 45(2):530–547, 2016. doi:10.1137/130938645.