ABSTRACT

Title of dissertation:	APPROXIMATION ALGORITHMS FOR FACILITY LOCATION AND CLUSTERING PROBLEMS
	Khoa Trinh, Doctor of Philosophy, 2017
Dissertation directed by:	Professor Aravind Srinivasan Department of Computer Science

Facility Location (FL) problems are among the most fundamental problems in combinatorial optimization. FL problems are also closely related to Clustering problems. Generally, we are given a set of facilities, a set of clients, and a symmetric distance metric on these facilities and clients. The goal is to "open" the "best" subset of facilities, subject to certain budget constraints, and connect all clients to the opened facilities so that some objective function of the connection costs is minimized. In this dissertation, we consider generalizations of classical FL problems. Since these problems are NP-hard, we aim to find good approximate solutions in polynomial time.

We study the classic k-median problem which asks to find a subset of at most k facilities such that the sum of connection costs of all clients to the closest facility is as small as possible. Our main result is a 2.675-approximation algorithm for this problem. We also consider the Knapsack Median (KM) problem which is a generalization of the k-median problem. In the KM problem, each facility is assigned an opening cost. A feasible set of opened facilities should have the total opening cost at most a given budget. The main technical challenge here is that the natural LP relaxation has unbounded integrality gap. We propose a 17.46-approximation algorithm for the KM problem. We also show that, after a preprocessing step, the integrality gap of the residual instance is bounded by a constant.

The next problem is a generalization of the k-center problem, which is called the Knapsack Center (KC) problem and has a similar budget constraint as in the KM problem. Here we want to minimize the maximum distance from any client to its closest opened facility. The KC problem is well-known to be 3-approximable. However, the current approximation algorithms for KC are deterministic and it is not hard to construct instances in which almost all clients have the worst-possible connection cost. Unfairness also arises in this context: certain clients may consistently get connected to distant centers. We design a randomized algorithm which guarantees that the expected connection cost of "most" clients will be at most $(1 + 2/e) \approx 1.74$ times the optimal radius and the worst-case distance remains the same. We also show a similar result for the k-center problem: all clients have expected approximation ratio about 1.592 with a deterministic upper-bound of 3 in the worst case.

It is well-known that a few *outliers* (very distant clients) may result in a very large optimal radius in the center-type problems. One way to deal with this issue is to cover only some t out of n clients in the so-called robust model. In this thesis, we give tight approximation algorithms for both robust k-center and robust matroid center problems. We also introduce a lottery model in which each client j

wants to be covered with probability at least $p_j \in [0,1]$. We then give randomized approximation algorithms for center-type problems in this model which match the worst-case bounds of the robust model and slightly violate the coverage and fairness constraints.

Several of our results for FL problems in this thesis rely on novel dependent rounding schemes. We develop these rounding techniques in the general setting and show that they guarantee new correlation properties. Given the wide applicability of the standard dependent rounding, we believe that our new techniques are of independent interests.

APPROXIMATION ALGORITHMS FOR FACILITY LOCATION AND CLUSTERING PROBLEMS

by

Khoa Trinh

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2017

To my grandfather Van-Chuc Tran,

who has boosted the love for Science in me.

Acknowledgments

I am grateful to receive tremendous help and support from many people during my PhD journey, whose names I would like to mention here.

First and foremost, I wholeheartedly thank my advisor Aravind Srinivasan for his constant encouragement and guidance that helped me overcome the most difficult times in my study. His wisdom and knowledge in mathematics and computer science have greatly inspired me. I will cherish all weekly meetings with Aravind, in which he would patiently listen to my problems and suggest many fruitful ideas. Aravind has also given me great advice on my career path, helped me find summer internships, and written recommendation letters when I was applying for a full-time job. I will remain eternally indebted to Aravind.

I would also like to thank my dear friend and colleague Tommy Pensyl for our collaboration over the past five years. We have spent countless hours together exploring and discussing various topics in AVW 3204. Indeed I have learnt a lot from Tommy. I really appreciate his time and valuable feedbacks to help me improve my presentation skills and prepare for many technical talks.

I have been very fortunate to collaborate with many amazing researchers. All chapters in this dissertation are coauthored with Aravind Srinivasan and Tommy Pensyl. Chapter 4 is also coauthored with Jarosław Byrka and Bartosz Rybicki. Chapters 5 and 6 are also coauthored with David Harris. I would also like to thank Bryan Lewis, Madhav Marathe, Pyrros A. Telionis, and Anil Vullikanti for our joint work on the ETU project.

I would like to thank Samir Khuller, David Mount, Mihai Pop, and Alexander Barg for being in my dissertation committee.

Many thanks to my labmates: Brian Brubach, Soheil Ehsani, Karthik Abinav Sankararaman, and Pan Xu for the good times, chit-chats, and lunches together. I especially want to thank Pan Xu for introducing me many wonderful Chinese and Korean restaurants. I am a little bit sad that we don't have any joint paper yet!

Throughout my PhD venture, I had three memorable internships at Oracle and Microsoft. I want to thank my managers and mentors who hosted me in the summers of 2014 to 2016, including Dmitry Krylov at Oracle, and Qifa Ke, Krishna Poola, Sreedal Menon, and Yajun Wang at Microsoft. I thank all of them for their mentorship and support.

My daily life in the US is so enjoyable thanks to my dear friends (most have lived in Graduate Hills): Bao Nguyen, Viet-An Nguyen, Lacey Yen Nguyen, My Le, Chanh Kieu, Sy Mai, Tuyet Le, Huong Vu, Bryan Dzung Ta, and Quang Huynh.

I also thank my parents Thu Tran and Hoa Trinh for their unconditional love, patience, and continued support during my study.

Finally, my research has been supported in part by the NSF Awards CNS 1010789 and CCF 1422569.

Table of Contents

List of Tables

List of Figures

List of Notations and Abbreviations

$\mathrm{cost}(j)$	the connection cost of client j
$\mathrm{cost}_{\mathcal{I}}(j)$	the connection cost of client j in instance $\mathcal{I}$
$\mathrm{cost}_{\mathcal{I}}(\mathcal{S})$	the cost of solution $\mathcal{S}$ for instance $\mathcal{I}$
$d(i,j)$	the distance from (client) j to (facility) i
$\mathrm{OPT}_{\mathcal{I}}$	the cost of an optimal, integral solution to instance $\mathcal{I}$
OPT_f	the cost of an optimal, fractional solution to the LP relaxation
KM	Knapsack Median
KC	Knapsack Center
MKC	Multi Knapsack Center
RkCenter	Robust k-center
RKnapCenter	Robust Knapsack Center
RMatCenter	Robust Matroid Center
FRkCenter	Fair Robust k-center
FRKnapCenter	Fair Robust Knapsack Center
FRMatCenter	Fair Robust Matroid Center

Chapter 1: Introduction

Combinatorial optimization is an important topic in Computer Science, which aims to find an optimal object from a finite (but possibly very large) set of objects. Many notable applications in other fields such as operations research, machine learning, artificial intelligence, and game theory involve solving discrete optimization problems. Unfortunately, most such optimization problems are NP-hard (see, e.g., [1–3].) Unless P=NP, which is believed to be highly unlikely by most researchers, there is no *efficient* algorithm (i.e., one that can find an optimal solution in polynomial time) for any NP-hard problem. The most popular approach in this case is to sacrifice the optimality and look for a good approximate solution. More formally, given an NP-hard minimization problem, we would like to find a polynomial-time algorithm which is provably guaranteed to return a solution whose cost is bounded by a small factor α times the optimal cost. We refer to such an algorithm as an α-approximation algorithm. The factor α is called the approximation ratio or approximation guarantee of the algorithm.

Our goal is to make the approximation ratio as small as possible. However, approximating an NP-hard problem to below certain threshold can also be NP-hard. An α-approximation algorithm for an NP-hard problem is said to be *tight* if there does not exist an $(\alpha - \epsilon)$-approximation algorithm for any $\epsilon > 0$, unless P=NP. Interestingly, not all NP-hard optimization problems are created equal: some problems can be significantly easier to approximate than others. For example, the Euclidean traveling salesman problem can be approximated to within a factor $(1+\epsilon)$ for any $\epsilon > 0$ [4] while it is NP-hard approximate the maximum maximum clique to a factor better than $O(n^{1-\gamma})$ for any $\gamma > 0$ [5].

The main focus of this dissertation is on designing good approximation algorithms for the class of facility-location problems. The rest of this chapter is organized as follows. Section 1 describes a somewhat general setting for facility-location problems and their applications. In Section 2, we shall review known results for several classic facility-location and clustering problems. Then we give an outline of this dissertation in Section 3.

1.1 Facility Location Problems

Facility Location (FL) problems are among the most fundamental problems in combinatorial optimization, which have been studied intensively and extensively in the past few decades. In the general setting, we are given a set of *facilities* $\mathcal{F}$, a set of *clients* $\mathcal{C}$, and a symmetric distance metric [1] d on $\mathcal{F} \cup \mathcal{C}$. That is, for any $i, j, k \in \mathcal{F} \cup \mathcal{C}$, we have $d(i, j) = d(j, i)$ and $d(i, j) + d(j, k) \geq d(i, k)$ (triangle inequality.) The goal is to open a subset of facilities, subject to certain feasibility constraints, and connect all clients to the open facilities so that some measurement

[1]Note that most of the Facility Location problems are NP-hard to approximate to within any constant factor under a general distance function. We only consider metric Facility Location problems in this dissertation.

of the connection cost and the facility opening cost is minimized. Three of the most well-studied, classic models under this setting are the Uncapacitated Facility Location (UFL) problem, the k-median problem, and the k-center problem.

In the UFL problem, each facility $i \in \mathcal{F}$ has an opening cost $f_i \in \mathbb{R}_+$ and our goal is to minimize the sum of the sum of connection cost from each client to the nearest open facility plus the total opening facility cost. On the other hand, in the k-median problem, we do not need to take into account the facility opening cost. Instead, we have a *cardinality constraint* saying that at most k facilities could be open. Moreover, if we change our objective to minimize the maximum distance from any client to the closest open facility and assume that $\mathcal{F} = \mathcal{C}$, then the problem is known as the k-center problem.

One natural generalization of the k-median and k-center problems is to replace the cardinality constraint by a knapsack constraint. That is, each facility i will now have a weight $w_i \geq 0$. We are given a *budget* $B > 0$ and require the total weight of the solution set to be at most B. Here the two problems are known as the Knapsack Median (KM) problem and Knapsack Center (KC) problem, respectively.

Another important way to define the feasibility constraint is requiring our solution to be an independent set of a given matroid $\mathcal{M}$. For example, in the Matroid Median problem, we look for a set of facilities being a basis of $\mathcal{M}$ which minimizes the total connection cost of all clients. Many other facility location problems, such as the data placement problem [6,7], the mobile facility location [8,9], the k-median forest problem [10], and the metric-uniform minimum-latency UFL problem [11], can be reduced to the Matroid Median problem [12].

FL problems have numerous applications in different areas, including Operational Research [13,14], Computational Biology [15], Computer Vision [16,17], Data Mining [18], and Network Design [19]. Depending on the context, we may need to use a variant of the basic models. For example, in certain cases, we may require that each facility $i \in \mathcal{F}$ can only serve at most $c_i > 0$ clients. This is called the *capacitated* version of the corresponding FL problem. In other applications, we may want to connect each client $j \in \mathcal{C}$ to exactly $r_j > 0$ facilities to ensure that j still gets connected even when $r_j - 1$ facilities serving j in the solution fail. This setting is called the *fault-tolerant* version of the original problem.

Another notable application of the FL problems is in the context of epidemic prevention and mitigation. Suppose we have a set of potential hospital locations and information about the population demands of some country, which could be just the number of people living at some location. We already have some models to predict the spread of some disease, e.g. Ebola, over all people across the country. Now we look for locations to place the new hospitals which are *most effective* in preventing/mitigating the disease. Then FL problems arise naturally in this context.

Facility Location problems are closely related to Clustering problems [20]. For example, the k-center problem can also be considered a Clustering problem, wherein the set of given points (which is $\mathcal{F} = \mathcal{C}$ in this case) is partitioned into at most k cluster with the objective function being the maximum cluster radius. Moreover, many techniques in FL problems have been successfully applied in other Clustering problems [21,22].

1.1.1 Summary

In summary, an instance of a FL problem studied in this thesis consists of a set of *facilities* $\mathcal{F}$, a set of *clients* $\mathcal{C}$, and a distance metric d on $\mathcal{F} \cup \mathcal{C}$. We want to find a set $\mathcal{S} \subseteq \mathcal{F}$ to minimize one of the following objective functions:

- the min-sum objective function: $\sum_{j \in \mathcal{C}} \min_{i \in \mathcal{S}} d(i,j)$,
- the min-max objective function: $\max_{j \in \mathcal{C}} \min_{i \in \mathcal{S}} d(i,j)$,

subject to one of the following feasibility constraints:

- the cardinality constraint: $|\mathcal{S}| \leq k$ for some given $k \in \mathbb{Z}_+$,
- the knapsack constraint: $\sum_{i \in \mathcal{S}} w_i \leq B$ for some given weight function $w : \mathcal{F} \to \mathbb{R}_+$ and $B \in \mathbb{R}_+$,
- the matroid constraint: $\mathcal{S}$ should be a basis of some given matroid $\mathcal{M}$.

1.2 Background and Related Work

Here we review the current approximability results for several classic FL problems.

1.2.1 FL problems with the min-sum objective function

The UFL problem is known to be NP-hard. Therefore, researchers focus on designing approximation algorithms which run in polynomial time and produce a solution whose cost can be bounded a (preferably small) constant factor times the optimal cost. Hochbaum [23] introduced an $O(\log n)$-approximation algorithm in 1982. Then Shmoys, Tardos and Aardal [24] gave the first constant approximation algorithm based on LP-rounding, where the ratio was 3.16. The approximation guarantee for UFL was then gradually improved by a series of papers [25–28]. The current best guarantee is 1.488 by the work of Shi Li [29] which carefully combines and randomizes JMS algorithm [30] and Byrka's algorithm [28]. On the negative side, Guha and Kuller [31] showed the first hardness result for UFL which states that, unless $\mathbf{NP} \subseteq \mathbf{DTIME}\left(n^{O(\log\log n)}\right)$, the UFL problem cannot be approximated to within a factor of 1.463. Then Sviridenko [26] improved the condition "$\mathbf{NP} \subseteq \mathbf{DTIME}\left(n^{O(\log\log n)}\right)$" to "$\mathbf{P} = \mathbf{NP}$".

The k-median problem is also known to be NP-hard. The first result for this problem was an $O(\log n \log\log n)$-approximation algorithm by Bartal [32]. Charikar, Guha, Tardos, and Shmoys [33] first gave a $6\frac{2}{3}$-approximation algorithm for k-median by an LP-rounding algorithm. Then, Jain and Vazirani [25] showed that one can use Lagrangian Relaxation to remove the budget constraint of opening at most k facilities so that k-median is reduced to a special case of the UFL problem. Next, they construct the so-called bi-point solution, losing a factor of 3 in the process. Finally, they round this bi-point solution to an integral feasible solution losing another multiplicative factor of 2, yielding a 6-approximation. Later, Jain, Mahdian, and Saberi (JMS [30]) improved the approximation ratio of constructing the bi-point solution to 2 by an improved greedy algorithm with a clever dual fitting analysis, resulting in a 4-approximation. Following this, Arya et. al. [34] introduced a local-search-based $(3+\epsilon)$-approximation algorithm.

Recently, Li and Svensson [35] give a breakthrough result stating that, given any α-approximate solution to the k-median problem using $k + O(1)$ facilities, one can still transform it into an $(\alpha + \epsilon)$-approximate feasible solution in polynomial time. By opening a small constant extra number of facilities, Li and Svensson manage to improve the ratio when rounding the bi-point solution from 3 to $\frac{1+\sqrt{3}}{2}$. Taking in account the factor of 2 when constructing the bi-point solution, this is a $(1 + \sqrt{3} + \epsilon) \approx (2.73 + \epsilon)$-approximation for the k-median problem. The current best known approximation guarantee is $(2.675 + \epsilon)$, given by Byrka et. al. [36] In this work, the authors carefully design a set of 9 different rounding strategies which altogether improve the factor lost when rounding the bi-point solution from $\frac{1+\sqrt{3}}{2} \approx 1.366$ to 1.337. On the negative side, Jain, Mahdian, and Saberi [30] showed that the k-median is **NP**-hard to approximate to within $1 + 2/e \approx 1.735$.

The Capacitated k-median (CKM) problem is notoriously difficult to approximate. Recall that, in this problem, each facility $i \in \mathcal{F}$ can only serve at most c_i different clients. There is no known constant approximation algorithm for this problem so far, nor do we know if such an algorithm exists. All known approximation algorithms so far either violate the capacitated constraint or the cardinality constraint. Byrka et. al. give an $O(1/\epsilon^2)$-approximation algorithm by violating the capacity constraint by a factor of $(2+\epsilon)$ in [37]. Recently, Shi Li [38] shows that there is an $O\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon}\right)$-approximation algorithm for the CKM problem if allowed to open $(1+\epsilon)k$ facilities and each facility may be open twice. A constant approximation ratio can also be achieved if allowed to violate the capacities by $(1+\epsilon)$ [39, 40].

On the other hand, the Capacitated facility location problem (CFL) is much easier to approximate. For the special case of uniform capacities, Korupolu et. al. [41] first gave a constant approximation, whose analysis was later improved by Chudak and Williamson [42]. The current best approximation ratio for the uniform CFL problem is 3 by Aggarwal et. al. [43]. For the general CFL problem, the first constant factor approximation algorithm was given by Pál et. al. [44]. Bansal et. al. [45] gave the current best 5-approximation algorithm for this problem. All mentioned algorithms are based on the local search method. The more recent work by An et. al. [46] introduced the first LP-based algorithm, settling the popular open question whether there exists such an algorithm for the CFL problem that was raised in [47]. Also, Svitkina [48] shows that the lower-bounded facility location problem, where each open facility should be assigned at least B clients, can be reduced to the CFL problem; and hence, admits a constant approximation guarantee.

The Fault-Tolerant k-median (FTKM) problem is also well-studied in the literature. As mentioned above, in this variant, we have to connect each client $j \in \mathcal{C}$ to at least $r_j \geq 1$ different facilities. When all r_j's are equal, Swamy and Shmoys [49] gave a 4-approximation by using the Lagrangian relaxation technique. Then Hajiaghayi et. al. [50] introduced a 93-approximation algorithm for the non-uniform case, which is currently the best known guarantee.

The Knapsack Median problem was first proposed by Krishnaswamy et. al. [51]. Although this is a generalization of the k-median problem, we still do not know a stronger lower-bound than $1 + 2/e$ for the KM problem. Krishnaswamy et. al. [51] gave a bicriteria $(16+\epsilon)$-approximation while violating the budget constraint by $(1 + \epsilon)$. Then Kumar [52] gave the first constant approximation algorithms for the problem although the ratio was very large – around 2700. Later, the work by Charikar & Li [53] and Swamy [12] improved the approximation ratio to 34 and 32, respectively. The main challenge when approximating this problem is due to the unbounded integrality gap of the natural LP relaxation. All approximation algorithms for the KM problem so far are using Kumar's bound. This bound allows us to upper-bound the connection cost of a cluster of nearby clients in terms of the cost of the optimal integral solution.

1.2.2 FL problems with the min-max objective function

One of the most basic and well-known problems in this class is the k-center problem. Recall that, in this problem, we have $\mathcal{F} = \mathcal{C}$ (i.e. each client is also a center) and want to minimize the maximum radius of all k clusters. The k-center problem is NP-hard and is 2-approximable [47]. Moreover, unless $\mathbf{P} = \mathbf{NP}$, we cannot approximate it to within factor $2 - \epsilon$ for any $\epsilon > 0$. If we require that some clients may not be open as a center (i.e. $\mathcal{F} \neq \mathcal{C}$), the problem is known as the k-supplier problem. It is relatively easy to modify most 2-approximation algorithms for k-center to obtain 3-approximation algorithms for the case $\mathcal{F} \neq \mathcal{C}$. Also, the lower-bound of k-supplier is known to be 3.

Similar to the KM problem, a natural generalization of the k-center problem is to assign a weight $w_j \geq 0$ to each client $j \in \mathcal{C}$ then require that the total weight of open centers should not exceed a given budget B. This problem is called the Knapsack Center problem and has a 3-approximation algorithm [54].

The Capacitated k-center problem seems to be easier to approximate than the CKM problem. It does admit a constant approximation ratio. Khuller and Sussmann [55] first gave a 6-approximation algorithm for the special case of uniform capacities. More than a decade later, the breakthrough work by Cygan et. al. [56] introduced the first constant factor approximation algorithm. Then An et. al. [57] improved the approximation ratio to 9, using a deterministic LP rounding algorithm. The Fault-Tolerant k-center with uniform requirements was considered by Sussmann and Khuller [58]. They gave a 3-approximation algorithm for the problem.

The Matroid Center problem is another generalization of the k-center problem. In this problem, we replace the cardinality constraint by a matroid constraint. That is, given matroid $\mathcal{M} = (\mathcal{F}, \mathcal{I})$ where $\mathcal{I}$ is the family of independent sets of the ground set $\mathcal{F}$, we require any feasible open set of centers to be a member of $\mathcal{I}$. This problem was first stuided by Chen et. al. [59], who also gave a (tight) 3-approximation algorithm.

In some cases, a few clients may cause the optimal radius of a k-center instance to be blown-up significantly. This issue was addressed by Charikar et. al. [60]. They referred to such clients as *outliers* and suggested a new model to deal with outliers. To this end, they introduced the so-called Robust k-center, in which we are also given an integer $t < n$ and only need to serve at least t clients in $\mathcal{C}$. Then they gave a 3-approximation algorithm for the Robust k-center problem. Chen et. al. [59] consider the Robust Matroid Center problem and provide a combinatorial algorithm which achieves a ratio of 7. Streaming and distributed algorithms for the k-center problem with outliers have been studied by McCutchen and Khuller [61] and by

Malkomes et. al. [62]. (Also, see [63] for a constant factor approximation algorithm for the k-median problem with outliers.)

Recently, there are new studies about generalizations of the k-center and k-supplier with outliers. The work by [64] considers the non-uniform k-center problem, in which we are given a set of radii $\{r_1, r_2, \ldots, r_k\}$ and want to assign these radii to the chosen k centers so that all points are covered. The authors show that the problem cannot be approximated to within any constant factor (unless P=NP) and give a bi-criteria approximation algorithm which achieves an $O(1)$-approximation and opens some $\Theta(1)$ centers of each radius. Also, Ahmadian and Swamy [65] give a 5-approximation algorithm for the k-supplier with outliers and with lower-bounds on the number of connections of each facility. Cygan and Kociumaka [66] show that there is a constant factor approximation for the Capacitated k-center problem with outliers.

1.2.3 Related Clustering problems

Depending on the applications, we may need different objective functions than the min-sum and min-max measurements as in k-median and k-center. A notable example is the k-sum-radii problem proposed by Charikar and Panigrahy [21]. In this problem, we want to partition a set of points in a metric space into k clusters so as to minimize the sum of cluster radii. Using similar ideas in [25] and [30], they applied the Lagrangian Relaxation technique to remove the cardinality constraint, used a primal-dual method to obtain a 3-approximation for the UFL-like problem, and then obtained a bi-point solution while only losing a factor $(1+\epsilon)$ in this process. After rounding the bi-point solution, their algorithm achieves an approximation ratio of ≈ 3.5. See [67–69] for more recent results on this problem.

Researchers have also tried to include additional requirements to obtain clusters satisfying desirable properties. Aggarwal et. al. [22] study some clustering problems in which we have a lower-bound on the number of points assigned to each cluster. For example, in the r-gather problem, we want to partition all given data points into clusters containing at least r points each so that the maximum radius of all clusters is minimized. Aggarwal et. al. [22] gave a (tight) 2-approximation algorithm for this problem. Clustering problems have also been studied under various models (see, e.g., [70–72]).

1.3 Dissertation outline

In this section, we briefly describe the our main technical contributions of the dissertation and how the next chapters are organized.

Chapter 2 discusses the dependent rounding technique which will be used extensively in the following chapters. It turns out that several of our FL problems require a new dependent-rounding method which can both preserve a set of hard "clustering contraints" and a set of soft knapsack constraints while guaranteeing "near-negative-correlation" among any subset of the variables. We will first review the basic dependent rounding scheme. Next, we describe a motivating problem in the context of FL problems and introduce new dependent rounding algorithms.

Finally, we prove the required correlation properties of our rounding schemes. Given the broad applicability of dependent rounding, we believe that the new techniques developed in this chapter will be of independent interests.

Chapter 3 presents an improved $(2.675+\epsilon)$-approximation algorithm for the k-median problem. We propose a set of 9 different randomized strategies to round the so-called *bi-point solution* of any k-median instance. By solving a non-linear factor-revealing program, we obtain an approximation ratio of 1.3371, which improves upon the ratio of $\frac{1+\sqrt{3}}{2} \approx 1.366$ by Li and Svensson [35]. Taking into account a factor of 2 lost due to construction of the *bi-point solution*, this gives a $(2.675+\epsilon)$-approximation algorithm. Using techniques developed in Chapter 2, we also improve the run-time from $n^{O(1/\epsilon^2)}$ to $n^{O((1/\epsilon)\log(1/\epsilon))}$.

In Chapter 4, we introduce a 17.46-approximation algorithm for the Knapsack Median problem, improving upon the 32-approximation algorithm by Swamy [12]. Our improvement in the approximation ratio comes from a new sparsification step which strengthens Kumar's bound, a simple clustering step inspired by [33], and randomization in the final rounding step. Our preprocessing algorithm yields the following by-product result: the LP relaxation of any *sparsed* instances has bounded integrality gap. We also give a bicriteria $(1+\sqrt{3}+\epsilon)$-approximation algorithm if allowed to violate the knapsack constraint by $(1+\epsilon)$.

We develop a new randomized algorithm for the Knapsack Center problem in Chapter 5, which not only matches the tight approximation ratio of 3 but also guarantees that almost all clients have expected connection cost at most $1+2/e \approx 1.735$ times the optimal radius. The two main technical ideas here are (i) applying a preprocessing step to ensure each center does not serve too many clients fractionally and (ii) utilizing the new dependent rounding method in Chapter 2.

Chapter 6 addresses the classic k-center problem. As mentioned before, there are simple 2-approximation algorithms for this problem. However, to the best of our knowledge, all of these algorithms are deterministic. For any such algorithm, it is not difficult to point out an instance in which most of the clients have connection cost actually matching the worst-case bound. Here we introduce a randomized algorithm which gives a slightly worse approximation guarantee of 3 but guarantees that all clients have expected approximation ratio only about 1.596.

Chapter 7 is about the center-type problems with outliers. We give tight approximation algorithms for the robust k-center and robust matroid center problems. We also introduce a new lottery model in which each client j requests a "target" probability p_j of being connected in the solution. Then we develop approximation algorithms for the *fair* robust k-center, *fair* robust knapsack center, and *fair* robust matroid center problems under this model.

Finally, we conclude the dissertation and discuss future works in Chapter 8.

Chapter 2: Dependent Rounding

2.1 Overview

2.1.1 Background

Randomized rounding is a very popular (and powerful) technique in designing approximation algorithms for NP-hard problems. The technique was first developed by Raghavan and Thompson [73]. The idea is to first solve the LP relaxation of our problem. Then the fractional LP solution $\mathbf{x} \in [0,1]^n$ will be rounded into an integral solution $\mathbf{X} \in \{0,1\}^n$ in a randomized manner. In the original work [73], each variable x_i will be rounded independently so that $\Pr[X_i = 1] = x_i$. This old technique has two main advantages. First, since the marginal probabilities are preserved, the expected "cost" of $\mathbf{X}$ will be equal to the optimal value of the LP by linearity of expectation. Secondly, one can apply Chernoff-type bounds on any linear function of $\mathbf{X}$ and argue that $\mathbf{X}$ is "feasible" with high probability.

In many applications, there are *hard* constraints which cannot be violated. For example, we may have a *cardinality* constraint: $\sum_{i=1}^{n} X_i \leq k$ for some parameter $k \in \mathbb{N}$. The technique by Raghavan and Thompson performs badly in this case as we have to "condition" on the event that the cardinality constraint is not violated.

In a seminal work, Ageev and Sviridenko [74] introduced a *deterministic* rounding scheme, called *pipage rounding*, which allows us to obtain a feasible $\mathbf{X}$ with probability one. Roughly speaking, the idea is to round $\mathbf{x}$ iteratively. In each step, we ensure that (i) at least one variable becomes integral and (ii) $\mathbf{x}$ is modified in such a way that its "cost" (normally a linear function of $\mathbf{x}$) does not increase.

The technique of Ageev and Sviridenko [74] was interpreted probabilistically in the work of Srinivasan [75] and further developed by Gandhi et. al. [76], giving rise to *dependent-rounding* schemes. In the past decade, dependent rounding techniques have found many applications in combinatorial optimization [53, 76–80]. Moreover, researchers have been trying to generalize the technique to make it work for matroid or matroid-intersection polytopes [81–83] and/or guarantee other correlation/concentration properties [36, 82, 84, 85].

The results by Srinivasan [75] are summarized in the following theorem.

Theorem 2.1.1 (Srinivasan [75]). *There exists an algorithm* $\textsc{DepRound}(\mathbf{x})$ *which takes as input a vector* $\mathbf{x} \in [0,1]^n$*, and outputs a vector* $\mathbf{X} \in \{0,1\}^n$ *in linear time with the following properties:*

(A1) *marginal probabilities are preserved:* $\Pr[X_i = 1] = x_i$*, for all* $i \in [n]$*,*

(A2) $\lfloor \sum_{i=1}^n x_i \rfloor \le \sum_{i=1}^n X_i \le \lceil \sum_{i=1}^n x_i \rceil$ *with probability one,*

(A3) *negative correlation: for any* $S \subseteq [n]$*, we have*

$$\Pr\left[\bigwedge_{i \in S} (X_i = 1)\right] \le \prod_{i \in S} x_i,$$

and

$$\Pr\left[\bigwedge_{i\in S}(X_i = 0)\right] \leq \prod_{i\in S}(1 - x_i).$$

The third property (A3) says that all variables are negatively correlated. In particular, this property allows us to apply Chernoff-type concentration bounds for linear functions of the variables [86].

2.1.2 Organization

Dependent rounding is a crucial tool to design (approximation) algorithms for facility-location problems in this thesis. In this chapter, we will discuss a new dependent-rounding technique, called *Symmetric Randomized Dependent Rounding* (SRDR), with stronger correlation properties. Both the basic dependent rounding and SRDR techniques will be applied extensively in later chapters. We also believe that SRDR is of independent interests.

The rest of this chapter is organized as follows. In Section 2, we review the weighted dependent rounding algorithm, which is slightly more general than the rounding scheme in [75]. In Section 3, we show that by randomly permuting the variables before applying weighted dependent rounding, the resulting variables will become "nearly" independent. Next, we formally state our basic problem (in the context of facility-location problems) and motivation for a new technique in Section 4. Then we consider a simple rounding algorithm which guarantees *half-negative* correlation among the variables in Section 5. Finally, we develop and analyze the

SRDR scheme in Section 6.

2.2 Weighted Dependent Rounding

Here we review the standard weighted dependent rounding algorithm. Suppose we are given a vector $\mathbf{x} \in [0,1]^n$ and a "weight" vector $\mathbf{a} \in \mathbb{R}^n_+$. The goal is to round $\mathbf{x}$ into an integral vector $\mathbf{X} \in \{0,1\}^n$ such that both the weighted sum and marginal probabilities are preserved (i.e., $\sum_{i=1}^n a_i X_i = \sum_{i=1}^n a_i x_i$ and $\mathbf{E}[X_i] = x_i$ for $i \in [n]$.) Note that this is always not possible: e.g., consider the case where there is only one variable $x_1 = 0.5$ with $a_1 = 1$. Thus, we may have to leave one fractional value in $\mathbf{X}$. Let $\mathrm{frac}(\mathbf{x}) = \{i \in [n] : 0 < x_i < 1\}$ be the set of indicies of fractional variables of $\mathbf{x}$. The basic algorithm is as follows.

Algorithm 1 SIMPLIFY$(\mathbf{x}, \mathbf{a})$

```
1: Choose any distinct pair $i^*, j^* \in \mathrm{frac}(\mathbf{x})$, $i^* \neq j^*$
2: Let $\delta_+ \leftarrow \min\{1 - x_{i^*}, (a_{j^*} x_{j^*})/a_{i^*}\}$
3: Let $\delta_- \leftarrow \min\{x_{i^*}, (a_{j^*}(1 - x_{j^*}))/a_{i^*}\}$
4: With probability $\frac{\delta_-}{\delta_+ + \delta_-}$,
       $$\mathbf{X} \leftarrow \mathbf{x} + (\delta_+)\mathbf{e}_{i^*} - \left(\frac{a_{i^*}}{a_{j^*}}\delta_+\right)\mathbf{e}_{j^*},$$
   else,
       $$\mathbf{X} \leftarrow \mathbf{x} - (\delta_-)\mathbf{e}_{i^*} + \left(\frac{a_{i^*}}{a_{j^*}}\delta_-\right)\mathbf{e}_{j^*}.$$
5: return X
```

Algorithm 2 WEIGHTEDDEPROUND$(\mathbf{x}, \mathbf{a})$

```
1: while $|\mathrm{frac}(\mathbf{x})| \geq 2$ do
2:     $\mathbf{x} \leftarrow$ SIMPLIFY$(\mathbf{x}, \mathbf{a})$
3: return x
```

To analyze WEIGHTEDDEPROUND, we need the following lemma.

Lemma 2.2.1. *Given any vector* $\mathbf{x} \in [0,1]^n$ *and* $\mathbf{a} \in \mathbb{R}_+^n$*. Suppose* $\mathbf{X}$ *is the output of* Simplify$(\mathbf{x}, \mathbf{a})$ *and* i^*, j^* *are indices chosen during the process. Then we have* $X_{i^*}, X_{j^*} \in [0,1]$ *and at least one of* X_{i^*}, X_{j^*} *is in* $\{0,1\}$*. Also, we have the following properties*

(B1) $a_{i^*}X_{i^*} + a_{j^*}X_{j^*} = a_{i^*}x_{i^*} + a_{j^*}x_{j^*}$,

(B2) $\mathbf{E}[X_{i^*}] = x_{i^*}$ *and* $\mathbf{E}[X_{j^*}] = x_{j^*}$,

(B3) $\mathbf{E}[X_{i^*}X_{j^*}] \leq x_{i^*}x_{j^*}$ *and* $\mathbf{E}[(1-X_{i^*})(1-X_{j^*})] \leq (1-x_{i^*})(1-x_{j^*})$.

Proof. By definition of δ_+ and δ_-, we have $x_{i^*}+\delta_+ \leq 1, x_{i^*}-\delta_- \geq 0, x_{j^*}-a_{i^*}\delta_+/a_{j^*} \geq 0$, and $x_{j^*} + a_{i^*}\delta_-/a_{j^*} \leq 1$. It is easy to verify that at least one of X_{i^*}, X_{j^*} is in $\{0,1\}$. If the first rule in line 4 is used, we have

$$\begin{aligned} a_{i^*}X_{i^*} + a_{j^*}X_{j^*} &= a_{i^*}(x_{i^*}+\delta_+) + a_{j^*}(x_{j^*} - a_{i^*}\delta_+/a_{j^*}) \\ &= a_{i^*}x_{i^*} + a_{j^*}x_{j^*}. \end{aligned}$$

Similarly, the second rule is called, we have

$$\begin{aligned} a_{i^*}X_{i^*} + a_{j^*}X_{j^*} &= a_{i^*}(x_{i^*}-\delta_-) + a_{j^*}(x_{j^*} + a_{i^*}\delta_-/a_{j^*}) \\ &= a_{i^*}x_{i^*} + a_{j^*}x_{j^*}. \end{aligned}$$

Thus, (B1) holds. By construction, we obtain

$$\begin{aligned}\mathbf{E}[X_{i^*}] &= \frac{\delta_-}{\delta_+ + \delta_-}(x_{i^*} + \delta_+) + \frac{\delta_+}{\delta_+ + \delta_-}(x_{i^*} - \delta_-) \\ &= x_{i^*}.\end{aligned}$$

Similarly, $\mathbf{E}[X_{j^*}] = x_{j^*}$. Finally, we have

$$\begin{aligned}\mathbf{E}[X_{i^*}X_{j^*}] &= \frac{\delta_-}{\delta_+ + \delta_-}(x_{i^*} + \delta_+)\left(x_{j^*} - \frac{a_{i^*}}{a_{j^*}}\delta_+\right) + \frac{\delta_+}{\delta_+ + \delta_-}(x_{i^*} - \delta_-)\left(x_{j^*} + \frac{a_{i^*}}{a_{j^*}}\delta_-\right) \\ &= x_{i^*}x_{j^*} - \frac{a_{i^*}}{a_{j^*}} \cdot \frac{\delta_+^2\delta_-}{\delta_+ + \delta_-} - \frac{a_{i^*}}{a_{j^*}} \cdot \frac{\delta_-^2\delta_+}{\delta_+ + \delta_-} \\ &\le x_{i^*}x_{j^*},\end{aligned}$$

since both δ_+ and δ_- are positive. Similarly, $\mathbf{E}[(1 - X_{i^*})(1 - X_{j^*})] \le (1 - x_{i^*})(1 - x_{j^*})$. □

Theorem 2.2.1. *Given any vector* $\mathbf{x} \in [0,1]^n$ *and* $\mathbf{a} \in \mathbb{R}^n_+$. *The algorithm* WEIGHTEDDEPROUND(

will return a (random) vector $\mathbf{X} \in [0,1]^n$ *with at most one floating value in* $(0,1)$ *in linear time such that*

(C1) *the weighted sum is preserved:* $\sum_{i=1}^n a_i X_i = \sum_{i=1}^n a_i x_i$ *with probability one,*

(C2) *the marginal probabilities are preserved:* $\mathbf{E}[X_i] = x_i$ *for all* $i \in [n]$,

(C3) *all variables are negatively correlated: for any* $S \subseteq [n]$, *we have*

$$\mathbf{E}\left[\prod_{i \in S} X_i\right] \le \prod_{i \in S} x_i,$$

and

$$\mathbf{E}\left[\prod_{i\in S}(1-X_i)\right] \le \prod_{i\in S}(1-x_i).$$

Proof. By Lemma 2.2.1, at least one new variable will become integral after each iteration of the while-loop at line 2. Thus, the algorithm terminates after $O(n)$ time and $\mathbf{X}$ has at most one remaining fractional value. Note that there are only two variables being changed in each iteration and their sum remains unchanged by Lemma 2.2.1. Then, by induction, (C1) holds.

Now fix any variable x_i. In each iteration, either x_i is not changed by SIMPLIFY or the expected value of the modified x_i remains the same. Again, by induction, $\mathbf{E}[X_i] = x_i$.

Finally, fix any $S \subseteq [n]$. For $k = 0, 1, 2, \ldots, n$, let $\mathbf{X}^{(k)}$ denote the (random) value of $\mathbf{x}$ after k steps. We will prove property (C3) by induction. When $k = 0$, the claim is vacuously true. For $k \ge 1$, suppose $\{i^*, j^*\}$ are indices chosen by SIMPLIFY in the k-th iteration. We consider the following cases:

- Case $S \cap \{i^*, j^*\} = \emptyset$:

$$\mathbf{E}\left[\prod_{i\in S} X_i^{(k)}\right] = \prod_{i\in S} X_i^{(k-1)},$$

- Case $S \cap \{i^*, j^*\} = \{i^*\}$:

$$\mathbf{E}\left[\prod_{i\in S} X_i^{(k)}\right] = \prod_{i\in S\setminus\{i^*\}} X_i^{(k-1)}\mathbf{E}\left[X_{i^*}^{(k)}\right] = \prod_{i\in S} X_i^{(k-1)},$$

- Case $S \cap \{i^*, j^*\} = \{j^*\}$:

$$\mathbf{E}\left[\prod_{i\in S} X_i^{(k)}\right] = \prod_{i\in S\setminus\{j^*\}} X_i^{(k-1)} \mathbf{E}\left[X_{j^*}^{(k)}\right] = \prod_{i\in S} X_i^{(k-1)},$$

- Case $S \cap \{i^*, j^*\} = \{i^*, j^*\}$:

$$\begin{aligned}\mathbf{E}\left[\prod_{i\in S} X_i^{(k)}\right] &= \prod_{i\in S\setminus\{i^*,j^*\}} X_i^{(k-1)} \mathbf{E}\left[X_{i^*}^{(k)} X_{j^*}^{(k)}\right] \\ &\leq \prod_{i\in S} X_i^{(k-1)}.\end{aligned}$$

Thus, $\mathbf{E}\left[\prod_{i\in S} X_i^{(k)}\right] \leq \prod_{i\in S} X_i^{(k-1)}$. Summing over all possible values of $\mathbf{X}^{(k-1)}$, we get

$$\begin{aligned}\mathbf{E}\left[\prod_{i\in S} X_i^{(k)}\right] &\leq \mathbf{E}\left[\prod_{i\in S} X_i^{(k-1)}\right] \\ &\leq \prod_{i\in S} x_i,\end{aligned}$$

by inductive hypothesis. Similarly, we have

$$\mathbf{E}\left[\prod_{i\in S}(1 - X_i)\right] \leq \prod_{i\in S}(1 - x_i).$$

$\square$

2.3 Near Independence via Random Permutation

The negative correlation property of WEIGHTEDDEPROUND gives us nice bounds of $\mathbf{E}\left[\prod_i X_i\right]$ and $\mathbf{E}\left[\prod_i (1 - X_i)\right]$. However, in certain cases, we may wish to bound products containing "mixed" factors of both X_i's and $(1 - X_i)$'s. (We will further discuss this issue in the next section.) Note that if all variables were rounded independently, the expected value of such products would be equal to the product of x_i's and $(1-x_i)$'s. While this property may not hold true in our dependent rounding algorithm, we may actually show that the expected value might not deviate too much from the target product as if the variables are independently rounded.

In this section, we consider a simple modification to the standard weighted dependent rounding scheme in Section 2.2 which yields the near-independence property. The main result is as follows.

Theorem 2.3.1. *Suppose we are given any vector* $\mathbf{x} \in [0, 1]^n$ *and* $\mathbf{a} \in \mathbb{R}^n_+$ *such that* $\min_i\{x_i, 1 - x_i\} \geq \alpha$ *and* $\frac{\max_i a_i}{\min_i a_i} \leq 1 + \alpha$ *for some constant* $\alpha \geq 0$*. There is an algorithm which can round* $\mathbf{x}$ *in linear time and return a (random) vector* $\mathbf{X} \in [0, 1]^n$ *with at most one floating value in* $(0, 1)$ *such that*

(D1) *the weighted sum is preserved:* $\sum_{i=1}^n a_i X_i = \sum_{i=1}^n a_i x_i$ *with probability one,*

(D2) *the marginal probabilities are preserved:* $\mathbf{E}[X_i] = x_i$ *for all* $i \in [n]$*,*

(D3) *any "small" subset of variables has the near-independence property: for any* *disjoint sets* $S, T \subseteq [n]$

$$\left(1 - \frac{8t(t-1)}{3n\alpha^2}\right)\lambda \leq \mathbf{E}\left[\prod_{i\in S} X_i \prod_{i \in T}(1 - X_i)\right] \leq \left(1 + \frac{8t}{3n\alpha^2}\right)^{t-1}\lambda,$$

where $t = |S \cup T|$.

2.3.1 Algorithm

The key idea is to permute the variables before applying the procedure SIMPLIFY on consecutive variables starting from the left.

Algorithm 3 WEIGHTEDDEPROUND2($\mathbf{x}, \mathbf{a}$)

```
1: Let $\pi$ be a random permutation of $(1, 2, \dots, n)$
2: $i^* \leftarrow \pi(1)$
3: for $j \leftarrow 2, 3, \dots, n$ do
4:     $j^* \leftarrow \pi(j)$
5:     $\mathbf{x} \leftarrow$ SIMPLIFY$(\mathbf{x}, \mathbf{a}, i^*, j^*)$
6:     if $x_j \in (0, 1)$ then
7:         $i^* \leftarrow j^*$
8: return $\mathbf{x}$
```

2.3.2 Analysis

In this section, we will prove that WEIGHTEDDEPROUND2 satisfies the properties stated in Theorem 2.3.1. Using similar arguments as in Section 2.2, it is not difficult to show the following lemma.

Lemma 2.3.1. *Given any vector* $\mathbf{x} \in [0, 1]^n$ *and* $\mathbf{a} \in \mathbb{R}^n_+$. *The algorithm* WEIGHTEDDEPROUND2($\mathbf{x}$ *will return a (random) vector* $\mathbf{X} \in [0, 1]^n$ *with at most one floating value in* $(0, 1)$ *in linear time such that*

(D1) *the weighted sum is preserved:* $\sum_{i=1}^n a_i X_i = \sum_{i=1}^n a_i x_i$ *with probability one,*

(D2) *the marginal probabilities are preserved:* $\mathbf{E}[X_i] = x_i$ *for all* $i \in [n]$.

So it remains to show the near-independence property. Let us take any $\mathbf{x} \in [0,1]^n$, $\mathbf{a} \in \mathbb{R}_+^n$ such that $\frac{\max_i a_i}{\min_i a_i} \leq 1+\alpha$, where $\alpha = \min_{i\in[n]}\{x_i, 1-x_i\}$. Let $\mathbf{X}$ be the resulting vector of WeightedDepRound2$(\mathbf{x}, \mathbf{a})$. For any disjoint sets $S, T \subseteq [n]$, we shall prove that

$$\left(1 - \frac{8t(t-1)}{3n\alpha^2}\right)\lambda \leq \mathbf{E}\left[\prod_{i\in S} X_i \prod_{i\in T}(1-X_i)\right] \leq \left(1 + \frac{8t}{3n\alpha^2}\right)^{t-1}\lambda, \tag{2.1}$$

where $t = |S \cup T|$ and $\lambda = \prod_{i\in S} x_i \prod_{i \in T}(1-x_i)$.

Suppose $S \cup T = \{j_1, j_2, \ldots, j_t\}$. Let $I := \{\pi^{-1}(j_1), \pi^{-1}(j_2), \ldots, \pi^{-1}(j_t)\}$ be the set of "positions" of the variables with respect to π. WLOG, assume that $I = \{i_1, \ldots, i_t\}$ where $i_1 < i_2 < \ldots < i_t$. For the ease of notation, we define $Y_j := X_{\pi(i_j)}$ if $\pi^{-1}(i_j) \in S$ and $Y_j := 1 - X_{\pi(i_j)}$ if $\pi^{-1}(i_j) \in S$ for $j = 1, 2, \ldots, t$. Also, define $q_j := \mathbf{E}[Y_j]$ for $j \in [t]$. Then, inequality (2.1) is equivalent to

$$\left(1 - \frac{8t(t-1)}{3n\alpha^2}\right)\prod_{i=1}^t q_i \leq \mathbf{E}\left[\prod_{i=1}^t Y_i\right] \leq \left(1 + \frac{8t}{3n\alpha^2}\right)^{t-1}\prod_{i=1}^t q_i. \tag{2.2}$$

In the rest of the analysis, we will analyze the change of $\mathbf{x}$ in the algorithm, but all the definitions of λ and q_j's so far should only depend on the original value of $\mathbf{x}$. Let $D_j = i_{j+1} - i_j$ for $j = 1, 2, \ldots, t-1$ and define $D_t = 1$. Note that D_j is a random variable which depends on π.

We say that a variable x_i is *fixed* if it is rounded to zero or one during the process. The key observation for this analysis is that, for any subset of t variables where t is small is enough, the distance between these variables in our permutation is relatively large. It means that there is a good chance that they will be fixed before being rounded together by Simplify. Intuitively, this will limit the positive correlation among the variables.

Lemma 2.3.2. *For any $j \in [t]$, let $\mathcal{E}_j$ denote the (bad) event that $x_{\pi(i_j)}$ is co-rounded with $x_{\pi(i_{j+1})}$ by* Simplify. *Define $\delta_j := (1-\alpha)^{(D_j-2)/2}$ for $j \le t-1$ and $\delta_t = 0$. We have that $\Pr[\mathcal{E}_j] \le \delta_j$ for all $j \in [t]$.*

Proof. The claim is vacuously true for $j = t$. Assume $j \le t-1$. By construction, $x_{\pi(i_j)}$ will be co-rounded with exactly $D_j - 1$ other variables before meeting $x_{\pi(i_{j+1})}$ if it does not get fixed. Thus, it suffices to prove that the probability that $x_{\pi(i_j)}$ is fixed in two consecutive calls of Simplify, say Simplify$(\mathbf{x}, \mathbf{a}, \pi(i_j), \pi(\ell))$ and Simplify$(\mathbf{x}, \mathbf{a}, \pi(i_j), \pi(\ell+1))$, where $\ell \in (j, t)$, is at least α, regardless of the current value of $x_{\pi(i_j)}$.

First, let us focus on the call of Simplify$(\mathbf{x}, \mathbf{a}, \pi(i_j), \pi(\ell))$. For the rest of this proof, for simplicity, let us just write x_{i_j} and a_ℓ to indicate $x_{\pi(i_j)}$ and $a_{\pi(\ell)}$. In this procedure, we set

$$\delta_+ = \min\left\{1 - x_{i_j}, \frac{a_\ell}{a_{i_j}} x_\ell\right\}, \quad \delta_- = \min\left\{x_{i_j}, \frac{a_\ell}{a_{i_j}}(1 - x_\ell)\right\}.$$

Then, with probability $\delta_-/(\delta_+ + \delta_-)$, we update $x_{i_j} \leftarrow x_{i_j} + \delta_+$. With remaining probability $\delta_+/(\delta_+ + \delta_-)$, we update $x_{i_j} \leftarrow x_{i_j} - \delta_-$. Consider the following cases:

- Case $1-x_{i_j} \leq \frac{a_\ell}{a_{i_j}} x_\ell$: we have $\delta_+ = 1-x_{i_j}$. If $\delta_- = x_{i_j}$ then x_{i_j} will be rounded to zero or one in both cases. Else, $\delta_- = \frac{a_\ell}{a_{i_j}}(1-x_\ell)$, then x_{i_j} will be equal to 1 with probability at least

$$\frac{\delta_-}{\delta_+ + \delta_-} = \frac{\frac{a_\ell}{a_{i_j}}(1-x_\ell)}{1-x_{i_j} + \frac{a_\ell}{a_{i_j}}(1-x_\ell)} \geq \frac{\frac{a_\ell}{a_{i_j}}(1-x_\ell)}{\frac{a_\ell}{a_{i_j}}x_\ell + \frac{a_\ell}{a_{i_j}}(1-x_\ell)} = 1 - x_\ell \geq \alpha.$$

- Case $1 - x_{i_j} > \frac{a_\ell}{a_{i_j}} x_\ell$: we have $\delta_+ = \frac{a_\ell}{a_{i_j}} x_\ell$. If $\delta_- = x_{i_j} \leq \frac{a_\ell}{a_{i_j}}(1-x_\ell)$, then x_{i_j} will be equal to 0 with probability at least

$$\frac{\delta_+}{\delta_+ + \delta_-} = \frac{\frac{a_\ell}{a_{i_j}}x_\ell}{\frac{a_\ell}{a_{i_j}}x_\ell + x_{i_j}} \geq \frac{\frac{a_\ell}{a_{i_j}}x_\ell}{\frac{a_\ell}{a_{i_j}}x_\ell + \frac{a_\ell}{a_{i_j}}(1-x_\ell)} \geq x_\ell \geq \alpha.$$

Thus, the only case that X_{i_j} does not get fixed by $\textsc{Simplify}(\mathbf{x}, \mathbf{a}, \pi(i_j), \pi(\ell))$ with probability at least α (in fact, x_{i_j} remains fractional with probability one in this case) is that

$$\frac{a_\ell}{a_{i_j}}(1-x_\ell) < x_{i_j} < 1 - \frac{a_\ell}{a_{i_j}} x_\ell.$$

Suppose this is the case and let x'_{i_j} denote the updated value of x_{i_j} after this step. Then the algorithm proceeds to call $\textsc{Simplify}(\mathbf{x}, \mathbf{a}, \pi(i_j), \pi(\ell+1))$. Using similar arguments, the only case that X_{i_j} does not get fixed by $\textsc{Simplify}(\mathbf{x}, \mathbf{a}, \pi(i_j), \pi(\ell+1))$ with probability at least α is that

$$\frac{a_{\ell+1}}{a_{i_j}}(1-x_{\ell+1}) < x'_{i_j} < 1 - \frac{a_{\ell+1}}{a_{i_j}} x_{\ell+1}. \tag{2.3}$$

Recall there are only two possible ways to update x_{i_j}:

- Case 1: With probability $\frac{\delta_-}{\delta_+ + \delta_-}$, we have

$$\begin{aligned} x'_{i_j} &= x_{i_j} + \delta_+ \\ &= x_{i_j} + \frac{a_\ell}{a_{i_j}} x_\ell \\ &> \frac{a_\ell}{a_{i_j}}(1 - x_\ell) + \frac{a_\ell}{a_{i_j}} x_\ell \\ &= \frac{a_\ell}{a_{i_j}} \geq \frac{1}{1+\alpha} = 1 - \frac{1}{1+\alpha}\alpha \geq 1 - \frac{a_{\ell+1}}{a_{i_j}} x_{\ell+1} > x'_{i_j}, \end{aligned}$$

  which is a contradiction.

- Case 2: With probability $\frac{\delta_+}{\delta_+ + \delta_-}$, we have

$$\begin{aligned} x'_{i_j} &= x_{i_j} - \delta_- \\ &= x_{i_j} - \frac{a_\ell}{a_{i_j}}(1 - x_\ell) \\ &< 1 - \frac{a_\ell}{a_{i_j}} x_\ell - \frac{a_\ell}{a_{i_j}}(1 - x_\ell) \\ &= 1 - \frac{a_\ell}{a_{i_j}} \leq 1 - \frac{1}{1+\alpha} = \frac{1}{1+\alpha}\alpha \leq \frac{a_{\ell+1}}{a_{i_j}}(1 - x_{\ell+1}) < x'_{i_j}, \end{aligned}$$

  which is also a contradiction.

Therefore, inequality (1) does not hold and x_{i_j} will get fixed in the second call with probability at least α. □

Lemma 2.3.3. *Conditioning on any fixed value of π, we have that*

$$\prod_{i=1}^{t} \max\{0, q_i - \delta_i\} \leq \mathbf{E}\left[\prod_{i=1}^{t} Y_i\right] \leq \prod_{i=1}^{t}(q_i + \delta_i).$$

Proof. Recall that q_i's are independent of π as dependent rounding always preserves the marginals regardless of the order of calling SIMPLIFY. We shall prove this claim by (backward) induction on the number of variables. For $k \in [t]$, we let $\mathcal{F}_k$ denote the an arbitrary, attainable configuration of the first k variables $Y_1, \ldots, Y_k$ that has been produced by the algorithm. Also, define $\mathcal{F}_0 := \emptyset$. We will prove that

$$\prod_{i=k}^{t} \max\{0, q_i - \delta_i\} \leq \mathbf{E}\left[\prod_{i=k}^{t} Y_i | \mathcal{F}_{k-1}\right] \leq \prod_{i=k}^{t} (q_i + \delta_i).$$

for all $k \in [t]$.

For the base case $k = t$, observe that $\mathbf{E}[Y_t|\mathcal{F}_{t-1}] = q_t$ and $\delta_t = 0$. Now, for any $k \leq t$, we can write

$$\begin{aligned}
&\mathbf{E}\left[\prod_{i=k}^{t} Y_i | \mathcal{F}_{k-1}\right] \\
&= \mathbf{E}\left[Y_i \prod_{i=k+1}^{t} Y_i | \mathcal{F}_{k-1}\right] \\
&= \mathbf{E}\left[Y_i \prod_{i=k+1}^{t} Y_i | \mathcal{E}_k \wedge \mathcal{F}_{k-1}\right] \Pr[\mathcal{E}_k|\mathcal{F}_{k-1}] + \mathbf{E}\left[Y_i \prod_{i=k+1}^{t} Y_i | \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}\right] \Pr[\bar{\mathcal{E}}_k|\mathcal{F}_{k-1}] \\
&= \mathbf{E}\left[Y_i \prod_{i=k+1}^{t} Y_i | \mathcal{E}_k \wedge \mathcal{F}_{k-1}\right] \Pr[\mathcal{E}_k] \\
&\qquad + \sum_y y \mathbf{E}\left[\prod_{i=k+1}^{t} Y_i | Y_k = y \wedge \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}\right] \Pr[Y_k = y | \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}] \Pr[\bar{\mathcal{E}}_k]. \qquad (2.4)
\end{aligned}$$

Upper-bound. We bound the two terms in (2.4) as follows.

$$\begin{aligned}\mathbf{E}\left[Y_i \prod_{i=k+1}^{t} Y_i | \mathcal{E}_k \wedge \mathcal{F}_{k-1}\right] \Pr[\mathcal{E}_k] &\leq \mathbf{E}\left[\prod_{i=k+1}^{t} Y_i | \mathcal{E}_k \wedge \mathcal{F}_{k-1}\right] \delta_k \\ &\leq \delta_k \prod_{i=k+1}^{t} (q_i + \delta_i),\end{aligned}$$

where the second inequality follows from the inductive hypothesis. Also,

$$\begin{aligned}&\sum_y y \mathbf{E}\left[\prod_{i=k+1}^{t} Y_i | Y_k = y \wedge \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}\right] \Pr[Y_k = y | \mathcal{F}_{k-1}] \Pr[\bar{\mathcal{E}}_k] \\ &\leq \prod_{i=k+1}^{t} (q_i + \delta_i) \sum_y y \Pr[Y_k = y | \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}] \Pr[\bar{\mathcal{E}}_k] \\ &= \prod_{i=k+1}^{t} (q_i + \delta_i) \sum_y y \Pr[Y_k = y \wedge \bar{\mathcal{E}}_k | \mathcal{F}_{k-1}] \\ &\leq \prod_{i=k+1}^{t} (q_i + \delta_i) \sum_y y \Pr[Y_k = y | \mathcal{F}_{k-1}] \\ &= \prod_{i=k+1}^{t} (q_i + \delta_i) \mathbf{E}[Y_k | \mathcal{F}_{k-1}] = q_k \prod_{i=k+1}^{t} (q_i + \delta_i),\end{aligned}$$

where the first inequality is due to inductive hypothesis. Plug the two above bounds into (2.4), we get

$$\mathbf{E}\left[\prod_{i=k}^{t} Y_i | \mathcal{F}_{k-1}\right] \leq \prod_{i=k}^{t} (q_i + \delta_i).$$

Lower-bound. From (2.4), we have

$$\begin{aligned}
&\mathbf{E}\left[\prod_{i=k}^{t} Y_i | \mathcal{F}_{k-1}\right] \\
&\geq \sum_y y \mathbf{E}\left[\prod_{i=k+1}^{t} Y_i | Y_k = y \wedge \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}\right] \Pr[Y_k = y | \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}] \Pr[\bar{\mathcal{E}}_k] \\
&\geq \prod_{i=k+1}^{t} \max\{0, q_i - \delta_i\} \sum_y y \Pr[Y_k = y | \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}] \Pr[\bar{\mathcal{E}}_k] \\
&= \prod_{i=k+1}^{t} \max\{0, q_i - \delta_i\} \mathbf{E}[Y_k | \bar{\mathcal{E}}_k \wedge \mathcal{F}_{k-1}] \Pr[\bar{\mathcal{E}}_k] \\
&= \prod_{i=k+1}^{t} \max\{0, q_i - \delta_i\} (\mathbf{E}[Y_k | \mathcal{F}_{k-1}] - \mathbf{E}[Y_k | \mathcal{E}_k \wedge \mathcal{F}_{k-1}] \Pr[\mathcal{E}_k]) \\
&\geq \prod_{i=k+1}^{t} \max\{0, q_i - \delta_i\} (q_k - \delta_k),
\end{aligned}$$

where we use inductive hypothesis for the first inequality and the fact that $\mathbf{E}[Y_k | \mathcal{E}_k \wedge \mathcal{F}_{k-1}] \Pr[\mathcal{E}_k] \leq \Pr[\mathcal{E}_k] = \delta_k$ in the final inequality. Since $\mathbf{E}\left[\prod_{i=k}^{t} Y_i | \mathcal{F}_{k-1}\right]$ is non-negative, we conclude that

$$\mathbf{E}\left[\prod_{i=k}^{t} Y_i | \mathcal{F}_{k-1}\right] \geq \prod_{i=k}^{t} \max\{0, q_i - \delta_i\}.$$

$\square$

Theorem 2.3.2 (Sandwich Theorem)**.** *We have*

$$\prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1} \max\left\{0, 1 - \frac{\delta_i}{q_i}\right\}\right] \leq \mathbf{E}\left[\prod_{i=1}^{t} Y_i\right] \leq \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1} \left(1 + \frac{\delta_i}{q_i}\right)\right].$$

Proof. Note that $\delta_t = 0$ by our definition. For any fixed π, Lemma 2.3.3 gives

$$\prod_{i=1}^{t} q_i \prod_{i=1}^{t-1} \max\left\{0, 1 - \frac{\delta_i}{q_i}\right\} \le \mathbf{E}\left[\prod_{i=1}^{t} Y_i\right] \le \prod_{i=1}^{t} q_i \prod_{i=1}^{t-1} \left(1 + \frac{\delta_i}{q_i}\right).$$

The claim follows by taking expectation over random choice of π. $\square$

The following technical lemmas will be useful later.

Lemma 2.3.4. *For any subset $J \subseteq [t-1]$ where $|J| = s \in [1, t]$, we have*

$$\mathbf{E}\left[\prod_{j \in J} \delta_j\right] = \mathbf{E}\left[\prod_{j \in [s]} \delta_j\right].$$

Proof. For any $S \subseteq [t-1]$, let us define

$$\begin{aligned} f(S) := \mathbf{E}\left[\prod_{j \in S} \delta_j\right] &= \mathbf{E}\left[\prod_{j \in S} (1-\alpha)^{(D_j - 2)/2}\right] \\ &= \mathbf{E}\left[(1-\alpha)^{(\sum_{j \in S} D_j - 2s)/2}\right] = \mathbf{E}\left[(1-\alpha)^{(D(S) - 2s)/2}\right]. \end{aligned}$$

Thus, to prove that $f(J) = f([s])$, it suffices to prove that $D(J)$ and $D([s])$ have the same distribution.

Recall that $I = \{i_1, i_2, \ldots, i_t\}$ is a random subset of $[n]$ and $D_j = i_{j+1} - i_j$ is the "distance" between element i_j and i_{j+1}. Let Ω denote the sample space of I (i.e., each event $I \in \Omega$ is a subset of size t of $[n]$.) For each integer $d \in [n]$, let $g(d, S, P) := |\{I \in P : D(S) = d\}|$. We claim that $g(d, J, \Omega) = g(d, [s], \Omega)$. To see this, note that Ω can be partitioned as $\Omega = \bigcup_{i,j \in [n]} \Omega_{ij}$ where Ω_{ij} is the collection of all subsets of size t with the first element being i and the last element being j. Then Ω_{ij} induces all possible sets $(D_1, D_2, \ldots, D_{t-1})$ such that $D_k \geq 1$ for all $k = 1, \ldots, t-1$ and $D_1 + \ldots + D_{t-1} = j - i - t + 1$, where each set is counted exactly once. Therefore, by symmetry, we have $g(d, J, \Omega_{ij}) = g(d, [s], \Omega_{ij})$. Summing over all Ω_{ij}'s gives $g(d, J, \Omega) = g(d, [s], \Omega)$.

Finally, for all $d \in [n]$, we have

$$\Pr[D(J) = d] = \frac{g(d, J, \Omega)}{\binom{n}{t}} = \frac{g(d, [s], \Omega)}{\binom{n}{t}} = \Pr[D([s]) = d],$$

which means that $D(J)$ and $D([s])$ have the same distribution. □

Claim 2.3.1. *For $0 \leq x < 1$ and $n \geq 0$, we have*

$$\sum_{k=0}^{\infty} \binom{k+n}{n} x^k = \frac{1}{(1-x)^{n+1}}.$$

Proof. Starting with the well-known series

$$1 + x + x^2 + \ldots = \frac{1}{1-x},$$

we take derivative of both sides n times and divide both sides by $n!$. □

Lemma 2.3.5. *For $s \leq t$, we have that*

$$\mathbf{E}\left[\prod_{j \in [s]} \delta_j\right] \leq \left(\frac{8t}{3n\alpha}\right)^s.$$

Proof. Let $D'_j := D_j - 1$ be the number of variables between $x_{\pi(i_j)}$ and $x_{\pi(i_{j+1})}$. Also, note that $\sqrt{1-\alpha} \le 1 - \alpha/2$. Then we have

$$\begin{aligned}
\mathbf{E}\left[\prod_{j\in[s]} \delta_j\right] &= \mathbf{E}\left[(1-\alpha)^{(D([s])-2s)/2}\right] \\
&= \mathbf{E}\left[(1-\alpha)^{(D'([s])-s)/2}\right] \\
&\le \mathbf{E}\left[(1-\alpha/2)^{D'([s])-s}\right] \\
&= \left(\frac{1}{1-\alpha/2}\right)^s \sum_{k=0}^{n-t} \Pr[D'_1 + \ldots + D'_s = k](1-\alpha/2)^k \\
&= \left(\frac{1}{1-\alpha/2}\right)^s \sum_{k=0}^{n-t} \frac{\binom{k+s-1}{s-1}\binom{n-k-s}{t-s}}{\binom{n}{t}}(1-\alpha/2)^k. \qquad (2.5)
\end{aligned}$$

Now observe that

$$\begin{aligned}
\frac{\binom{n-k-s}{t-s}}{\binom{n}{t}} &= \frac{t!(n-k-s)!(n-t)!}{(t-s)!n!(n-k-t)!} \\
&= \frac{(t-s+1)\ldots t}{(n-s+1)\ldots n} \times \frac{(n-k-t+1)\ldots(n-t)}{(n-k-s+1)\ldots(n-s)} \\
&\le \left(\frac{t}{n}\right)^s.
\end{aligned}$$

Plugging this into 2.5, we obtain

$$\begin{aligned}
\mathbf{E}\left[\prod_{j\in[s]} \delta_j\right] &\le \left(\frac{t}{n(1-\alpha/2)}\right)^s \sum_{k=0}^{n-t} \binom{k+s-1}{s-1}(1-\alpha/2)^k \\
&\le \left(\frac{t}{n(1-\alpha/2)(\alpha/2)}\right)^s = \left(\frac{2t}{n\alpha(1-\alpha/2)}\right)^s \le \left(\frac{8t}{3n\alpha}\right)^s.
\end{aligned}$$

where the penultimate inequality is due to Claim 2.3.1 and the final inequality follows from the fact that $\alpha \le 1/2$. $\square$

We are now ready to prove the required bound (2.2). For the lower-bound, we shall apply Weierstrass inequality which says that $\prod_i (1 - x_i) \ge 1 - \sum_i x_i$ where $x_i \le 1$ for all i. By Theorem 2.3.2,

$$\begin{aligned}
\mathbf{E}\left[\prod_{i=1}^{t} Y_i\right] &\ge \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1} \max\left\{0, 1 - \frac{\delta_i}{q_i}\right\}\right] \\
&\ge \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1} \max\left\{0, 1 - \frac{\delta_i}{\alpha}\right\}\right] \\
&= \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1} \left(1 - \min\left\{1, \frac{\delta_i}{\alpha}\right\}\right)\right] \\
&\ge \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[1 - \sum_{i=1}^{t-1} \min\left\{1, \frac{\delta_i}{\alpha}\right\}\right] \\
&\ge \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[1 - \sum_{i=1}^{t-1} \frac{\delta_i}{\alpha}\right] \\
&= \prod_{i=1}^{t} q_i \cdot \left(1 - \frac{1}{\alpha} \sum_{i=1}^{t-1} \mathbf{E}[\delta_i]\right) \\
&= \prod_{i=1}^{t} q_i \cdot \left(1 - \frac{t-1}{\alpha} \mathbf{E}[\delta_1]\right) \ge \prod_{i=1}^{t} q_i \cdot \left(1 - \frac{8t(t-1)}{3n\alpha^2}\right),
\end{aligned}$$

where the penultimate equality follows from Lemma 2.3.4 and the last inequality is due to Lemma 2.3.5.

For the upper-bound, we shall expand the product, apply Lemma 2.3.5 for each term $\mathbf{E}[\prod \delta_j]$ and then collapse them together using the binomial theorem. By

Theorem 2.3.2,

$$\begin{aligned}
\mathbf{E}\left[\prod_{i=1}^{t} Y_i\right] &\le \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1}\left(1+\frac{\delta_i}{q_i}\right)\right] \\
&\le \prod_{i=1}^{t} q_i \cdot \mathbf{E}\left[\prod_{i=1}^{t-1}\left(1+\frac{\delta_i}{\alpha}\right)\right] \\
&= \prod_{i=1}^{t} q_i \cdot \left(1+\sum_{k=1}^{t-1}\binom{t}{k}\frac{\mathbf{E}\left[\prod_{j\in[k]}\delta_j\right]}{\alpha^k}\right) \\
&\le \prod_{i=1}^{t} q_i \cdot \left(1+\sum_{k=1}^{t-1}\binom{t}{k}\left(\frac{8t}{3n\alpha^2}\right)^k\right) \\
&= \prod_{i=1}^{t} q_i \cdot \left(1+\frac{8t}{3n\alpha^2}\right)^{t-1}.
\end{aligned}$$

This concludes the proof of Theorem 2.3.1.

2.4 A Motivating Problem

The main focus of this thesis is on facility-location and clustering problems. In particular, we study LP rounding algorithms this class of problems. In general, such methods usually have two main steps. The first step, often called the *filtering* step, is to assign each "center" or "facility" to the closest "client". The set of client j and all facilities associated with it is called a *cluster*, denoted as F_j. We also refer to j as the cluster center of F_j. By the LP constraints, there should be at least one (fractionally) opened facility inside F_j. We then "filter-out" some clusters to obtain a maximal collection $\mathcal{F}'$ of pairwise-disjoint clusters. Note that the clusters removed in this step must intersect with some cluster in $\mathcal{F}'$; and hence, one can bound the distance from their cluster centers to the cluster centers in $\mathcal{F}'$. The second step involves rounding the fractional solution subject to a budget constraint (or a matroid constraint, depending on our problem) and a set of *cluster constraints*. The idea is to pick exactly one facility inside each cluster $F_j \in \mathcal{F}'$ to open. This strategy guarantees that (i) each cluster center in $\mathcal{F}'$ can be served effectively by a facility in its cluster and (ii) other clients can be connected "indirectly" to the facilities serving such cluster centers. Indeed, dependent rounding is a natural choice for this task.

We now formulate our basic problem. Suppose we have m disjoint sets $F_1, \ldots, F_m \subseteq [n]$, a vector $\mathbf{a} \in \mathbb{R}^n_+$ for some integer $n > 0$, and a parameter $B > 0$. Consider the following polytope

$$\mathcal{P} = \left\{ \mathbf{v} \in [0,1]^n : v(F_j) = 1 \;\; \forall j \in [m]; \quad \sum_{i=1}^{n} a_i v_i \leq B. \right\}$$

We are given a point $\mathbf{x} \in \mathcal{P}$ and want to round $\mathbf{x}$ into an integral point $\mathbf{X}$ inside $\mathcal{P}$. (Here $x_i \in (0,1)$ is the "extent" that we want to pick facility i.) As mentioned before, $\mathcal{P}$ needs not contain any integral points. Fortunately, in our applications, it suffices to obtain an "almost" integral point with a few remaining fractional values.

It is not difficult to show that all extreme points of $\mathcal{P}$ have at most two fractional value. One can simply decompose $\mathbf{x}$ into a convex combination of $t \leq n+1$ extreme points of $\mathcal{P}$:

$$\mathbf{x} = p_1 \mathbf{x}_1 + p_2 \mathbf{x}_2 + \ldots + p_t \mathbf{x}_t,$$

where $p_1, \ldots, p_t \geq 0$ and $\sum_{i=1}^{t} p_i = 1$. Now if we randomly pick $\mathbf{X} := \mathbf{x}_i$ with probability p_i, we have that $\mathbf{E}[X_j] = x_j$ for all $j \in [n]$. In some applications (e.g., the Knapsack Median problem in Chapter 3), preserving the marginal probabilities is all we need to obtain the desired results. In many other cases, we may require *negative-correlation* between the variables.

To see the importance of the *negative-correlation* property in our context, consider any client j which is not a cluster center in $\mathcal{F}'$. The LP constraint ensures that $x(F_j) = 1$. If all x_i's are negative correlated, we have that

$$\begin{aligned}\Pr[\text{no center in } F_j \text{ is opened}] &= \mathbf{E}\left[\prod_{i \in F_j}(1 - x_i)\right] \\ &\leq \prod_{i \in F_j}(1 - x_i) \\ &\leq \prod_{i \in F_j} e^{-x_i} = 1/e.\end{aligned}$$

Thus, with probability at least $1 - 1/e$, client j can be connected to some open inside its cluster F_j. Only with probability $\leq 1/e$, do we need to connect it indirectly via another cluster center in $\mathcal{F}'$. Similar ideas have been exploited to design approximation algorithms for various FL problems (see, e.g., [28, 53, 78, 87]).

In this chapter, we will discuss two approaches to our basic problem. Here we will try to describe these methods as general-purpose rounding algorithms because we believe that they are of independent interests.

Approach 1. One key ingredient which makes negative correlation possible in weighted dependent rounding is that we only change two variables at a time: one is increased and the other is decreased. As we have extra cluster constraints, this is not always possible anymore. For example, suppose $n = 2, a_1 = 1, a_2 = 2, x_1 = x_2 = 0.5$, and $B = 1.5$. If we also require $x_1 + x_2 = 1$, then $\mathbf{x}$ is already uniquely defined by two constraints ($x_1 + x_2 = 1$ and $x_1 + 2x_2 = 1.5$). In this case, we cannot change $\mathbf{x}$ without violating the constraints. This is not surprised as we already know that an extreme point of $\mathcal{P}$ may have two fractional values.

On the other hand, as long as $\mathbf{x}$ contains at least 3 floating variables, we still have some freedom to round it. Consider the following cases:

- Case 1: if there exists a cluster F_j with at least 3 fractional variables, say x_1, x_2, and x_3, then it is easy to find $\vec{z} = (z_1, z_2, z_3)$ such that, for any $\delta \in \mathbb{R}$,

  $$(x_1 + \delta z_1) + (x_2 + \delta z_2) + (x_3 + \delta z_3) = x_1 + x_2 + x_3,$$

  and

  $$a_1(x_1 + \delta z_1) + a_2(x_2 + \delta z_2) + a_3(x_3 + \delta z_3) = a_1x_1 + a_2x_2 + a_3x_3.$$

  Thus, we can change these variables, either along $\vec{z}$ or $-\vec{z}$, until one of them becomes zero or one.

- Case 2: if none of the clusters has ≥ 3 fractional values, there must be two clusters F_j and $F_{j'}$ such that each of them contains 2 floating variables. Suppose F_j contains $x_1, x_2 \in (0, 1)$ and $F_{j'}$ contains $x_3, x_4 \in (0, 1)$. Again, in this case, we can find $\vec{z} = (z_1, z_2, z_3, z_4)$ such that, for any $\delta \in \mathbb{R}$, we have

$$\begin{aligned}
(x_1 + \delta z_1) + (x_2 + \delta z_2) &= x_1 + x_2, \\
(x_3 + \delta z_3) + (x_4 + \delta z_4) &= x_3 + x_4, \\
a_1(x_1 + \delta z_1) + a_2(x_2 + \delta z_2) + a_3(x_3 + \delta z_3) + a_4(x_4 + \delta z_4) &= a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4.
\end{aligned}$$

(The vector $\vec{z}$ exists as we have 4 variables while there are only 3 constraints.)

Again, we can update $\mathbf{x}$ along $\vec{z}$ or $-\vec{z}$ until at least one more variable gets rounded to zero or one.

Observe that there will be exactly two variables which are both increased or decreased together in the first case. Similarly, in the second case, rounding $\mathbf{x}$ by $\vec{z}$ will result in exactly two variables increased and two variables decreased. Thus, we may have two variables moving in the same direction in a single step. This means that the variables are not negatively correlated anymore. However, as the number variables that are *positively* correlated in a single step is still small (two in this case), we can still prove the following useful property: "all variables x_i are *half*-negatively correlated." That is, for any $S \subseteq [n]$, we have

$$\mathbf{E}\left[\prod_{i \in S} x_i\right] \leq \sqrt{\prod_{i \in S} x_i}, \qquad \text{and} \qquad \mathbf{E}\left[\prod_{i \in S}(1 - x_i)\right] \leq \sqrt{\prod_{i \in S}(1 - x_i)}.$$

In Section 4, we will prove a generalized theorem which states that if there are at most s variables changing in the same direction at a time, then we still have *partial* negative correlation among the variables. More formally, for any $S \subseteq [n]$,

$$\mathbf{E}\left[\prod_{i \in S} x_i\right] \le \prod_{i \in S} x_i^{1/s}, \qquad \text{and} \qquad \mathbf{E}\left[\prod_{i \in S}(1 - x_i)\right] \le \prod_{i \in S}(1 - x_i)^{1/s}.$$

Approach 2. In the previous approach, we see how to obtain *half*-negative correlation property by carefully choosing the variables to co-round in each iteration. In particular, this leads to a weaker bound on our bad event:

$$\begin{aligned}\Pr[\text{no center in } F_j \text{ is opened}] &= \mathbf{E}\left[\prod_{i \in F_j}(1 - x_i)\right] \\ &\le \sqrt{\prod_{i \in F_j}(1 - x_i)} \le \sqrt{\prod_{i \in F_j} e^{-x_i}} = e^{-1/2},\end{aligned}$$

which is much worse than $1/e$. Can we still improve this bound? The answer is yes but with an additional condition: the vector $\mathbf{X}$ may now contain some $O(1)$ left-over fractional values.

First, for each cluster $F_k \in \mathcal{F}'$ let $C_k := F_j \cap F_k$ denote the set of facilities in F_k that j is interested in. Let Y_k be the indicator for the event that some facility in C_k is opened. The probability that no center in F_j is open is now equal to $\mathbf{E}\left[\prod_k (1 - Y_k)\right]$. Now observe that the updating rule in Case 1 of Approach 1 will process only one cluster $F_k \in \mathcal{F}$ at a time. So the random variables Y_k are pairwise independent in this case. (While the variables x_i's are not negatively correlated, this fact actually does not affect our bad event.) Thus, we can repeatedly simplify $\mathbf{x}$ by this rule until each cluster F_k has exactly two fractional variables, say $x_{i_1(k)}$ and $x_{i_2(k)}$. Also, assume that $a_{i_1(k)} \leq a_{i_2(k)}$.

Now we shall not use the second rule in Approach 1 to round $\mathbf{x}$ as it would introduce positive correlation to Y_k's. Instead, for each cluster F_k, let us define Z_k to be the indicator that $Z_k = 1$ if $i_2(k)$ is open and $Z_k = 0$ if $i_1(k)$ is open. The problem is now reduced to obtaining such a (random) vector Z that $\sum_k Z_k a'_k \leq B'$ where $a'_k = a_{i_2(k)} - a_{i_1(k)}$ and $B' = B - \sum_k a_{i_1(k)}$ and $\mathbf{E}[Z_k] = z_k := x_{i_2(k)}$. Unfortunately, we cannot simply use the standard weighted dependent rounding to round $\mathbf{z}$. Although the variable Z_k's will be negatively correlated, this property is not enough to give a good bound on $\mathbf{E}\left[\prod_k (1 - Y_k)\right]$.

Let us analyze $\mathbf{E}\left[\prod_k (1 - Y_k)\right]$. WLOG, we may assume that C_k consists of either $i_1(k)$ or $i_2(k)$ but not both of them. (If none of them is in C_k, the factor $(1 - Y_k)$ has no contribution to the expression. If both of them are in C_k, then one of them will be opened and the bad event does not happen.) Let $S := \{k : i_1(k) \in C_k\}$ and $T := \{k : i_2(k) \in C_k\}$. We have

$$\mathbf{E}\left[\prod_k (1 - Y_k)\right] = \mathbf{E}\left[\prod_{k \in S} Z_k \prod_{k \in T} (1 - Z_k)\right]. \tag{2.6}$$

So we need a good upper-bound on the RHS of (2.6). The main technical difficulty here is that we have "mixed" terms of Z_k and $(1 - Z_k)$ in the product. In Section 5, we will introduce a new dependent rounding technique, called *Symmetric Randomized*

Dependent Rounding (SRDR), which guarantees the following correlation:

$$\mathbf{E}\left[\prod_{k\in S} Z_k \prod_{k\in T}(1-Z_k)\right] \le \left(\prod_{k\in S} z_k \prod_{k\in T}(1-z_k)\right)^{1-1/(t+1)},$$

where t is the number of remaining unrounded values in $\mathbf{Z}$. In fact, setting $t = O(1/\epsilon)$ suffices to obtain an upper-bound of $(1/e+\epsilon)$ in our problem. Also, the left-over $2t$ fractional values in $\mathbf{X}$ can be rounded deterministically after a pre-processing step. See Chapter 5 for further details.

2.5 Partial Negative Correlation

In this section, we study a more general setting in which a vector $\mathbf{x} \in [0,1]^n$ is (randomly) rounded into an "almost" integral vector subject to multiple linear constraints. The main idea is trying to keep the maximum number s of variables which are updated in the same direction as small as possible in every single step. Then we can show a trade-off between s and the "amount" of negative correlation.

Specifically, suppose we are given a polytope $\mathcal{P} = \{\mathbf{v} \in [0,1]^n : A\mathbf{v} \le \mathbf{b}\}$ where $A \in \mathbb{R}^{m\times n}$ is an $m \times n$ matrix and $\mathbf{b} \in \mathbb{R}^n$. In most applications, the number of constraints m is (much) smaller than the number of variables n. Observe that any extreme point of $\mathcal{P}$ should have at most m fractional values in general. Then the resulting vector may contain up to m fractional values. Here suppose we aim to round $\mathbf{x}$ until it contains at most some $t \ge 0$ remaining fractional values. The actual choice of t may depend on the structure $\mathcal{P}$. For example, we might choose $t = 2$ in the motivating problem discussed in Section 3.

2.5.1 Algorithm

The rounding algorithm is as follows.

Algorithm 4 Simplify2($\mathbf{x}, A, \mathbf{b}$)

```
1: Let $R = \{i \in [m] : A_i\mathbf{x} = b_i \wedge \exists j \in [n] : x_j \notin \{0,1\}, A_{ij} \neq 0\}$ be the set of tight
   constraints containing at least one floating variable.
2: Let $A'$ be the sub-matrix of $A$ containing only rows in $R$.
3: Let $\mathbf{r} \in \mathbb{R}^n$ be such that
     • $\mathbf{r} \neq 0$ and $r_i = 0$ for all $i \in [n] : x_i \in \{0,1\}$,
     • $A'\mathbf{r} = 0$.
4: Let $\delta_+, \delta_- > 0$ be such that
     • $\mathbf{x} + \delta_+\mathbf{r} \in [0,1]^n$ and $\mathbf{x} - \delta_-\mathbf{r} \in [0,1]^n$,
     • either $\mathbf{x} + \delta_+\mathbf{r}$ has one more integral value or there exists $i' \notin R$ such that
       $A_{i'}(\mathbf{x} + \delta_+\mathbf{r}) = b_{i'}$ and $A_i(\mathbf{x} + \delta_+\mathbf{r}) \leq b_i$ for all $i \neq i'$,
     • either $\mathbf{x} - \delta_-\mathbf{r}$ has one more integral value or there exists $i'' \notin R$ such that
       $A_{i''}(\mathbf{x} - \delta_-\mathbf{r}) = b_{i''}$ and $A_i(\mathbf{x} - \delta_-\mathbf{r}) \leq b_i$ for all $i \neq i''$.
5: With probability $\frac{\delta_-}{\delta_+ + \delta_-}$,
       $$\mathbf{X} \leftarrow \mathbf{x} + \delta_+\mathbf{r},$$
   else,
       $$\mathbf{X} \leftarrow \mathbf{x} - \delta_-\mathbf{r}.$$
6: return X
```

Algorithm 5 GeneralDepRound($\mathbf{x}, A, \mathbf{b}, t$)

```
1: while $|\text{frac}(\mathbf{x})| > t$ do
2:     $\mathbf{x} \leftarrow$ Simplify2($\mathbf{x}, \mathbf{a}$)
3: return x
```

Discussion. We note that the vector $\mathbf{r}$ in line 2 of Simplify2 exists as long as $t \geq |R|$ (or, more precisely, the number of floating variables is strictly greater than the row rank of A'.) In addition, we would prefer a direction $\mathbf{r}$ that has as few elements with the same sign as possible because such a vector will limit positive correlation among changing variables in the process. Next, the magnitudes δ_+, δ_- are chosen so that when moving $\mathbf{x}$ along $\mathbf{r}$ and $-\mathbf{r}$, we either get one more rounded variable or hit a new tight constraint. (Observe that both δ_+ and δ_- are finite.) Therefore, the algorithm GeneralDepRound should terminate after at most $m + n$ iterations. Our result is summarized in the following theorem.

Theorem 2.5.1. *Suppose we are given a matrix $A \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^n$, $\mathcal{P} = \{\mathbf{v} \in [0,1]^n : A\mathbf{v} \leq \mathbf{b}\}$, and a point $\mathbf{x} \in \mathcal{P}$. Let t be any integer greater than or equal to the maximum number of fractional values of any extreme point of $\mathcal{P}$. Then* GeneralDepRound$(\mathbf{x}, A, \mathbf{b}, t)$ *will return a random vector $\mathbf{X}$ in $O(m+n)$ time such that*

(C1) *$\mathbf{X}$ contains at most t fractional values,*

(C2) *$\mathbf{X} \in \mathcal{P}$,*

(C3) *$\mathbf{E}[X_i] = x_i$ for all $i \in [n]$,*

(C4) *Assume the number of non-zero elements in* r *which have the same sign is at most s whenever* Simplify2 *is executed. Then, for any $S \subseteq [n]$, we have that*

$$\mathbf{E}\left[\prod_{i \in S} X_i\right] \leq \mathbf{E}\left[\prod_{i \in S} X_i^{1/s}\right] \leq \prod_{i \in S} x_i^{1/s}, \tag{2.7}$$

and

$$\mathbf{E}\left[\prod_{i \in S}(1 - X_i)\right] \leq \mathbf{E}\left[\prod_{i \in S}(1 - X_i)^{1/s}\right] \leq \prod_{i \in S}(1 - x_i)^{1/s}. \tag{2.8}$$

2.5.2 Analysis

Here we prove Theorem 2.5.1. By the choice of t, whenever the procedure SIMPLIFY2 is called, $\mathbf{x}$ is not an extreme point of $\mathcal{P}$. It means that the currently floating variables are not uniquely defined by A'; and hence, there must exist a vector $\mathbf{r}$ satisfying properties in line 3. Again, the choice of δ_+, δ_- and the update rules in line 5 ensure that either we have a new tight constraint or $\mathbf{x}$ has a new rounded variable. Thus, the running time of GENERALDEPROUND is $O(m+n)$. Property (C1) satisfied by the condition of the while-loop in line 1.

For property (C2), we prove the invariant that if $\mathbf{x} \in \mathcal{P}$ then the resulting vector by SIMPLIFY2 is also in $\mathcal{P}$. Recall that the output of SIMPLIFY2 is either $\mathbf{x}+\delta_+\mathbf{r}$ or $\mathbf{x}-\delta_-\mathbf{r}$. These two vectors are in $[0,1]^n$ by our choice of δ_+, δ_-. Moreover, by definition of $\mathbf{r}$, we have $A(\mathbf{x}+\delta_+\mathbf{r}) = A\mathbf{x} + \delta_+ A\mathbf{r} = A\mathbf{x} \le \mathbf{b}$. Similarly, $A(\mathbf{x} - \delta_-\mathbf{r}) \le \mathbf{b}$.

Assuming $\mathbf{x}' := \text{SIMPLIFY2}(\mathbf{x}, \mathbf{a})$, we claim that $\mathbf{E}[x'_i] = x_i$ for all $i \in [n]$. Then property (C3) follows by induction. By construction, we have

$$\mathbf{E}[x'_i] = \frac{\delta_-}{\delta_+ + \delta_-}(x_i + \delta_+ r_i) + \frac{\delta_+}{\delta_+ + \delta_-}(x_i - \delta_- r_i) = x_i.$$

To prove property (C4), we first need the following lemma.

Lemma 2.5.1. *The procedure* SIMPLIFY2$(\mathbf{x}, \mathbf{a})$ *will return a vector* $\mathbf{X}$ *such that, for any* $S \subseteq [n]$, *we have* $\mathbf{E}\left[\prod_{i\in S} X_i^{1/s}\right] \le \prod_{i \in S} x_i^{1/s}$.

Proof. Let $T := \{i \in [n] : r_i \neq 0\}$ be the set of indices of floating variables which are being changed by the procedure. Also, let $S' := S \cap T$. Define $g(\delta) := \prod_{i \in S'}(x_i + \delta r_i)$ and $f(\delta) := g(\delta)^{1/s}$ be functions of $\delta \in (-\delta_-, \delta_+)$. We first prove that $f(\delta)$ is concave on its domain. The first derivative of g is

$$\begin{aligned} g'(\delta) &= \sum_{i \in S'} r_i \prod_{j \in S' \setminus \{i\}} (x_i + \delta r_i) \\ &= \sum_{i \in S'} r_i \frac{g(\delta)}{x_i + \delta r_i} = g(\delta) \sum_{i \in S'} A_i, \end{aligned}$$

where we define $A_i := \frac{r_i}{x_i + \delta r_i}$. By definition of δ_- and δ_+, the denominator of A_i is in $(0, 1)$ for all $\delta \in (-\delta_-, \delta_+)$; and hence, A_i is well-defined. Next, we have

$$\begin{aligned} g''(\delta) &= \sum_{i \in S'} r_i \left(\prod_{j \in S' \setminus \{i\}} (x_i + \delta r_i) \right)' \\ &= \sum_{i \in S'} r_i \left(\sum_{j \in S' \setminus \{i\}} r_j \prod_{k \in S' \setminus \{i,j\}} (x_k + \delta r_k) \right) \\ &= \sum_{i \in S'} r_i \left(\sum_{j \in S' \setminus \{i\}} r_j \frac{g(\delta)}{(x_i + \delta r_i)(x_j + \delta r_j)} \right) \\ &= g(\delta) \sum_{i \in S'} A_i \sum_{j \in S' \setminus \{i\}} A_j \\ &= 2g(\delta) \sum_{i,j \in S' : i < j} A_i A_j. \end{aligned}$$

Now note that $f'(\delta) = \frac{1}{s} g(\delta)^{1/s-1} g'(\delta)$, and

$$\begin{aligned}
f''(\delta) &= \left(\frac{1}{s} g(\delta)^{1/s-1} g'(\delta)\right)' \\
&= \frac{1}{s}\left(g(\delta)^{1/s-1}\right)' g'(\delta) + \frac{1}{s} g(\delta)^{1/s-1} g''(\delta) \\
&= \frac{1}{s}\left((1/s-1) g(\delta)^{1/s-2} g'(\delta)^2 + g(\delta)^{1/s-1} g''(\delta)\right) \\
&= \frac{g(\delta)^{1/s-2}}{s}\left(\frac{1-s}{s} g'(\delta)^2 + g(\delta) g''(\delta)\right) \\
&= \frac{1}{g(\delta)^{2-1/s} s^2}\left((1-s) g'(\delta)^2 + s g(\delta) g''(\delta)\right) \\
&= \frac{1}{g(\delta)^{2-1/s} s^2}\left((1-s)\left(g(\delta)\sum_{i \in S'} A_i\right)^2 + s g(\delta)\left(2 g(\delta) \sum_{i,j \in S' : i<j} A_i A_j\right)\right) \\
&= \frac{g(\delta)^{1/s}}{s^2}\left((1-s)\left(\sum_{i \in S'} A_i\right)^2 + 2s\left(\sum_{i,j \in S' : i<j} A_i A_j\right)\right).
\end{aligned}$$

We will show that $f''(\delta) < 0$ for $\delta \in (-\delta_-, \delta_+)$. Let $S^+ := \{i \in S' : r_i > 0\}$ and

$S^- := \{i \in S' : r_i < 0\}$. Recall that $|S^+|, |S^-| \le s$ and $S' = S^+ \cup S^-$. Then we have

$$\begin{aligned}
& (1-s)\left(\sum_{i\in S'} A_i\right)^2 + 2s\left(\sum_{i,j\in S':i<j} A_iA_j\right) \\
= & -(s-1)\sum_{i\in S'} A_i^2 + 2\left(\sum_{i,j\in S':i<j} A_iA_j\right) \\
= & -(s-1)\left(\sum_{i\in S^+} A_i^2 + \sum_{i\in S^-} A_i^2\right) + 2\left(\sum_{i,j\in S':i<j} A_iA_j\right) \\
\le & -(|S^+|-1)\sum_{i\in S^+} A_i^2 - (|S^-|-1)\sum_{i\in S^-} A_i^2 + 2\left(\sum_{i,j\in S':i<j} A_iA_j\right) \\
= & -\sum_{i,j\in S:i<j,r_ir_j>0} (A_i^2 - 2A_iA_j + A_j^2) + 2\left(\sum_{i,j\in S:i<j,r_ir_j<0} A_iA_j\right) \\
= & -\sum_{i,j\in S:i<j,r_ir_j>0} (A_i - A_j)^2 + 2\left(\sum_{i,j\in S:i<j,r_ir_j<0} A_iA_j\right) \\
< & \ 0,
\end{aligned}$$

where the last inequality follows because the sign of A_i is the same as the sign of r_i for all i, implying that the second sum is negative. So f is concave on $(-\delta_-, \delta_+)$. Moreover, since f is right-continuous at $-\delta_-$ and left-continuous at δ_+, we have

$$\begin{aligned}
\frac{\delta_-}{\delta_+ + \delta_-} f(\delta_+) + \frac{\delta_+}{\delta_+ + \delta_-} f(-\delta_-) &\le f\left(\frac{\delta_-}{\delta_+ + \delta_-}\delta_+ + \frac{\delta_+}{\delta_+ + \delta_-}(-\delta_-)\right) \\
&= f(0).
\end{aligned}$$

Finally, we obtain

$$\begin{aligned}
\mathbf{E}\left[\prod_{i\in S} X_i^{1/s}\right] &= \prod_{i\in S\setminus S'} x_i^{1/s}\mathbf{E}\left[\prod_{i\in S'} X_i^{1/s}\right] \\
&= \prod_{i\in S\setminus S'} x_i^{1/s}\left(\frac{\delta_-}{\delta_+ + \delta_-} f(\delta_+) + \frac{\delta_+}{\delta_+ + \delta_-} f(-\delta_-)\right) \\
&\le \prod_{i\in S\setminus S'} x_i^{1/s} f(0) = \prod_{i\in S} x_i^{1/s}.
\end{aligned}$$

□

Now property (C4) follows easily by induction. Let $\mathbf{X}^{(k)}$ be the value of $\mathbf{x}$ after k steps. We will show that $\mathbf{E}\left[\prod_{i\in S}\left(X_i^{(k)}\right)^{1/s}\right] \le \prod_{i\in S} x_i^{1/s}$. The base case when $k=0$ is vacuously true. For $k\ge 1$ and a fixed value of $\mathbf{X}^{(k-1)}$, by Lemma 2.5.1, we have

$$\mathbf{E}\left[\prod_{i\in S}\left(X_i^{(k)}\right)^{1/s}\right] \le \prod_{i\in S}\left(X_i^{(k-1)}\right)^{1/s}.$$

Summing over all possible values of $\mathbf{X}^{(k-1)}$ and by inductive hypothesis, we get

$$\mathbf{E}\left[\prod_{i\in S}\left(X_i^{(k)}\right)^{1/s}\right] \le \mathbf{E}\left[\prod_{i\in S}\left(X_i^{(k-1)}\right)^{1/s}\right] \le \prod_{i\in S} x_i^{1/s}.$$

We have thus proved (2.7). The proof of inequality (2.8) is very similar and is omitted here.

2.6 Symmetric Randomized Dependent Rounding

In this section, we introduce the SRDR technique. Given any "weight" vector $\mathbf{a} \in (\mathbb{R} \setminus \{0\})^n$, a fractional vector $\mathbf{x} \in [0,1]^n$, and a parameter t, the technique allows us to efficiently round $\mathbf{x}$ into an "almost" integral vector $\mathbf{X}$ – one with at most t fractional values left. Like in standard dependent rounding [75], the expected value of the X_i's and the weighted sum are preserved: $\mathbf{E}[X_i] = x_i$ for all $i \in [n]$ and $\sum a_i x_i = \sum a_i X_i$. Moreover, any subset of the variables has the following strong property:

$$\mathbf{E}\left[\Big(\prod_{i \in S} X_i \prod_{i \in T} (1 - X_i)\Big)^p\right] \le \left(\prod_{i \in S} x_i \prod_{i \in T} (1 - x_i)\right)^p, \tag{2.9}$$

where $p = 1 - 1/(t+1)$, for any $S, T \subseteq [n]$ and $S \cap T = \emptyset$.

Note that if all X_i's are independent, then the equality holds for all p. Also, the more variables remain fractional, the closer p is to one. Intuitively, the "amount of dependence" is proportional to the number of times $\mathbf{x}$ is "simplified" in the rounding algorithm. By introducing an "early" stopping condition, the product in the LHS gets closer to what it would have been if all the variables are independent. In this case, the variables are said to be *nearly-independent.*

We have showed the near-independence property of dependent rounding in Section 2.3. Recall that the idea is to randomly permute the vector $\mathbf{x}$ before applying the dependent rounding of [75]. Then any two variables are "far" from each other and unlikely to be rounded together in a single round, which implies that any small groups of the variables are nearly independent. Our results here are different in several ways. First, our upper-bound (2.9) only depends on the remaining number of fractional variables (i.e., is independent of the number of terms n and of the ratio $a_{\max}/a_{\min}$), and the "target" probabilities need not be bounded away from 0 or 1. Secondly, it does not require the weights to be non-negative as in WeightedDepRound2.

In each iteration of the standard dependent-rounding technique of [75], we will co-round two variables, say x_1 and x_2, in such a way that the sum $a_1x_1 + a_2x_2$ is preserved and the expected values of x_1, x_2 do not change. If $a_1a_2 > 0$, then an increase in x_1 will lead to a decrease in x_2 for the sum to remain the same, and vice versa. This explains why we obtain negative correlation for all techniques in [36, 75, 76, 83]. Now suppose the weights can be arbitrary and $a_1a_2 < 0$. In this case, to preserve the sum, if we increase x_1, we must also increase x_2, and vice versa. Thus, we may have some positive correlation between the variables. Our SRDR technique employs the following ideas to reduce the "amount of" positive correlation. First, we randomly pick a pair of (x_1, x_2) to co-round in each iteration so that the probability that $a_1a_2 < 0$ is only about the 1/2 in worst case. Next, instead of enforcing either x_1 or x_2 to be integral after a single step, we allow both x_1 and x_2 to remain fractional in some cases. For example, suppose $a_1 = +1, a_2 = -1, x_1 = 0.1$, and $x_2 = 0.2$. The normal approach will round (x_1, x_2) into $(0, 0.1)$ with probability 8/9 and $(0.9, 1)$ with probability 1/9. Our idea is to round this pair "symmetrically", getting $(0, 0.1)$ with probability 1/2 and $(0.2, 0.3)$ with probability 1/2; importantly, this symmetric amount of change (a parameter called δ in SRDR) is chosen globally, instead of on the particular a_i, x_i chosen (at random). All of these make our analyses of SRDR possible.

The main results of this section are summarized in the following theorem.

Theorem 2.6.1 (Upper-bound)**.** *Given vectors $\mathbf{x} \in [0,1]^n$, $\mathbf{a} = (a_1, \ldots, a_n) \in (\mathbb{R} \setminus \{0\})^n$, and $t \in \mathbb{N}$, there exists a randomized algorithm $\mathcal{A}$ which can round $\mathbf{x}$ in expected $O(n^3)$ time and return a vector $\mathbf{X} \in [0,1]^n$ with at most t fractional values. Both the weighted sum and and all the marginal probabilities are preserved: $\sum_i a_i X_i = \sum_i a_i x_i$ and $\mathbf{E}[X_i] = x_i$ for all $i \in [n]$. Moreover, for any disjoint sets $S, T \subseteq [n]$, we have*

$$\mathbf{E}\left[\Big(\prod_{i \in S} X_i \prod_{i \in T}(1 - X_i)\Big)^p\right] \le \left(\prod_{i \in S} x_i \prod_{i \in T}(1 - x_i)\right)^p,$$

where $p = 1 - 1/(t+1)$. In addition, for n and t sufficiently large, we have

$$\mathbf{E}\left[\prod_{i \in S} X_i \prod_{i \in T}(1 - X_i)\right] \le \left(\prod_{i \in S} x_i \prod_{i \in T}(1 - x_i)\right)\left(1 + \frac{m \log(1/\alpha)}{t}\right),$$

where $m = |S \cup T|$ and $\alpha = \min_i\{x_i, 1 - x_i\}$.

2.6.1 Algorithm

Suppose we are given vectors $\mathbf{x} \in [0,1]^n$ and $\mathbf{a} = (a_1, \ldots, a_n) \in (\mathbb{R} \setminus \{0\})^n$. The algorithm to round $\mathbf{x}$ is as follows.

Algorithm 6 SRSimplify$(\mathbf{x}, \mathbf{a})$

```
1: $\delta \leftarrow \min_{i \in \mathrm{frac}(\mathbf{x})} \{|a_i| x_i, |a_i|(1 - x_i)\}$
2: Randomly choose a pair $i^*, j^* \in \mathrm{frac}(\mathbf{x})$ where $i^* < j^*$
3: With probability 1/2,
       $\mathbf{X} \leftarrow \mathbf{x} + (\delta / a_{i^*}) \mathbf{e}_{i^*} - (\delta / a_{j^*}) \mathbf{e}_{j^*}$,
   else,
       $\mathbf{X} \leftarrow \mathbf{x} - (\delta / a_{i^*}) \mathbf{e}_{i^*} + (\delta / a_{j^*}) \mathbf{e}_{j^*}$.
4: return X
```

Algorithm 7 SRDR$(\mathbf{x}, \mathbf{a}, t)$

```
1: while $|\mathrm{frac}(\mathbf{x})| > t$ do
2:     $\mathbf{x} \leftarrow$ SRSimplify$(\mathbf{x}, \mathbf{a})$
3: return x
```

2.6.2 Analysis

Lemma 2.6.1. *Given vectors $\mathbf{x} \in [0,1]^n$, $\mathbf{a} = (a_1, \ldots, a_n) \in (\mathbb{R} \setminus \{0\})^n$, and $t \in \mathbb{N}$, the algorithm* SRDR *will return a vector $X \in [0,1]^n$ with at most t fractional values in expected $O(n^3)$ time. Moreover, the weighted sum and marginal probabilities are both preserved: $\sum_i a_i X_i = \sum_i a_i x_i$ and $\mathbf{E}[X_i] = x_i$ for all i.*

Proof. Observe that, in the procedure SRSimplify$(\mathbf{x}, \mathbf{a})$, δ is chosen to be the maximum value such that all possible values of $\mathbf{X}$ in step 3 of SRSimplify$(\mathbf{x}, \mathbf{a})$ remain in $[0,1]^n$. At least one of the $\binom{|\mathrm{frac}(\mathbf{x})|}{2}$ choices of (i^*, j^*) will lead to at least 1 new integral element in step 3 of SRSimplify. Thus, with probability at least $\frac{2}{|\mathrm{frac}(\mathbf{x})|^2} \geq \frac{2}{n^2}$, it fixes an element in a single round. In expectation, we need $O(n^3)$ iterations to round all n variables. By construction, the weighted sum $\sum_i a_i x_i$ and all marginal probabilities are preserved after every single round. Then

$\sum_i a_i X_i = \sum_i a_i x_i$ and $\mathbf{E}[X_i] = x_i$ for all i by induction. $\square$

Next, we prove the following inequality.

Lemma 2.6.2. *For all integers $k \geq 1$, $n \geq 2$, and any $p \in [0,1]$ define $f_k(n,p) := n\binom{2p}{2k} + (n-2)\binom{p}{k}(-1)^k$. We have that $f_k(n, 1-1/n) \leq 0$.*

Proof. We will prove this lemma by induction. For $k = 1$, we have

$$\begin{aligned} f_1(n, 1-1/n) &= n\frac{2(1-1/n)(2(1-1/n)-1)}{2} - (n-2)(1-1/n) \\ &= (1-1/n)(n(2-2/n-1) - n + 2) = 0. \end{aligned}$$

For $k \geq 2$, we have

$$\begin{aligned} &\left(1 - \frac{p+1}{k}\right)\left(f_{k-1}(n,p) - n\frac{2p(k-p-1)}{k(2k-1)}\binom{2p}{2(k-1)}\right) \\ &= \frac{k-p-1}{k}\left(n\binom{2p}{2(k-1)} + (n-2)\binom{p}{k-1}(-1)^{k-1} - n\frac{2p(k-p-1)}{k(2k-1)}\binom{2p}{2(k-1)}\right) \\ &= n\binom{2p}{2(k-1)}\left(\frac{k-p-1}{k} - \frac{2p(k-p-1)}{k(2k-1)}\right) + (n-2)\binom{p}{k-1}\frac{p-k+1}{k}(-1)^k \\ &= f_k(n,p). \end{aligned}$$

It follows that $f_k(n,p) \leq 0$ when $k \geq 2, n \geq 1$ and $p = 1 - 1/n$ as $\frac{p+1}{k} \leq 1$, $\frac{2p(k-p-1)}{k(2k-1)}\binom{2p}{2(k-1)} \geq 0$, and $f_{k-1}(n,p) \leq 0$. $\square$

Lemma 2.6.3. *Let $\mathbf{X}$ be the output of* SRSimplify$(\mathbf{x}, \mathbf{a})$ *for some vectors $\mathbf{x} \in (0,1)^n$, $\mathbf{a} \in (\mathbb{R} \setminus \{0\})^n$, and $n \geq 2$. For any set $S \subseteq [n]$,*

$$\mathbf{E}\left[\Big(\prod_{i \in S} X_i\Big)^p\right] \leq \Big(\prod_{i \in S} x_i\Big)^p$$

holds for $p = 1 - \frac{1}{n}$. Furthermore, if all weights in $\{a\}_{i \in S}$ have the same sign, the inequality holds for $p = 1$.

Proof. WLOG, assume that all elements of $\mathbf{x}$ are floating, i.e. $|\text{frac}(\mathbf{x})| = n$. Fix a set $S \in [n]$. We will analyze the expected value of $\Lambda := \prod_{i \in S} X_i$ in terms of $\lambda := \prod_{i \in S} x_i$. Define

$$b_i := \begin{cases} 1/a_i & i \in S \\ \\ 0 & i \notin S \end{cases}, \qquad A_i := 1 + \frac{b_i \delta}{x_i}, \qquad B_i := 1 - \frac{b_i \delta}{x_i}.$$

Then the expected value of Λ^p conditioned on particular (i^*, j^*) being chosen is:

$$\begin{aligned} \mathbf{E}[\Lambda^p | (i^*, j^*) = (i,j)] &= \frac{1}{2}\lambda^p \left(\frac{x_i + b_i\delta}{x_i}\right)^p \left(\frac{x_j - b_j\delta}{x_j}\right)^p + \frac{1}{2}\lambda^p \left(\frac{x_i - b_i\delta}{x_i}\right)^p \left(\frac{x_j + b_j\delta}{x_j}\right)^p \\ &= \frac{1}{2}\lambda^p (A_i^p B_j^p + B_i^p A_j^p). \end{aligned}$$

Observe that $\mathbf{E}[\Lambda | (i^*, j^*) = (i,j)] = \lambda \left(1 - \frac{b_i b_j \delta^2}{x_i x_j}\right)$, and if b_i *and* b_j have the same sign (or one or both are 0), then this quantity is at most λ. Thus, if *all* of $\{a_i\}_{i \in S}$ have the same sign, SRDR preserves full negative correlation: $\mathbf{E}[\Lambda] \leq \lambda$. (This is equivalent to the known negative correlation property of the standard positively-weighted dependent rounding scheme.)

We now continue with the general case. What the proof boils down to is expanding all terms of the form $(1 + x)^p$ using the generalized binomial theorem, multiplying everything to get a polynomial in x, and then showing (in Lemma 2.6.2) that $p = 1 - \frac{1}{n}$ is the precise value which causes the higher order terms to vanish (and the lower order terms to be negative). The algebra is much cleaner if we first manipulate the equation before doing any expansions. In the following summations, $i, j \in [n]$. We have

$$\begin{aligned}
\mathbf{E}[\Lambda^p] &= \sum_{i<j} \Pr[\{i^*, j^*\} = \{i, j\}]\mathbf{E}[\Lambda^p | \{i^*, j^*\} = \{i, j\}] \\
&= \frac{1}{\binom{n}{2}} \cdot \sum_{i<j} \frac{1}{2}\lambda^p(A_i^p B_j^p + B_i^p A_j^p) \\
&= \frac{\lambda^p}{4\binom{n}{2}} \cdot \sum_{i<j} \left((A_i^p + B_i^p)(A_j^p + B_j^p) - (A_i^p - B_i^p)(A_j^p - B_j^p)\right) \\
&= \frac{\lambda^p}{8\binom{n}{2}} \cdot \left(\left(\sum_i (A_i^p + B_i^p)\right)^2 - \sum_i (A_i^p + B_i^p)^2 - \left(\sum_i (A_i^p - B_i^p)\right)^2 + \sum_i (A_i^p - B_i^p)^2\right) \\
&\le \frac{\lambda^p}{8\binom{n}{2}} \cdot \left(n \sum_i (A_i^p + B_i^p)^2 - \sum_i (A_i^p + B_i^p)^2 + \sum_i (A_i^p - B_i^p)^2\right) \\
&= \frac{\lambda^p}{8\binom{n}{2}} \cdot \sum_i \left(n(A_i^{2p} + B_i^{2p}) + 2(n-2)A_i^p B_i^p\right).
\end{aligned}$$

To get the penultimate inequality we applied the Cauchy-Schwarz inequality and the fact that any square is nonnegative. We will show that for the appropriate choice of p, $\mathbf{E}[\Lambda^p] \le \lambda^p$. We now expand A_i and B_i using the generalized binomial theorem.

(Note that $\frac{b_i\delta}{x_i} \in [-1,1]$ by our choice of δ.)

$$\begin{aligned}
\mathbf{E}[\Lambda^p] &\le \frac{\lambda^p}{8\binom{n}{2}} \cdot \sum_i \left(n\left(1+\frac{b_i\delta}{x_i}\right)^{2p} + n\left(1-\frac{b_i\delta}{x_i}\right)^{2p} + 2(n-2)\left(1-\left(\frac{b_i\delta}{x_i}\right)^2\right)^p \right) \\
&= \frac{\lambda^p}{8\binom{n}{2}} \cdot \sum_i \Bigg(n\sum_{k\ge 0}\binom{2p}{k}\Big(\frac{b_i\delta}{x_i}\Big)^k + n\sum_{k\ge 0}\binom{2p}{k}(-1)^k\Big(\frac{b_i\delta}{x_i}\Big)^k + \\
&\qquad\qquad 2(n-2)\sum_{k\ge 0}\binom{p}{k}(-1)^k\Big(\frac{b_i\delta}{x_i}\Big)^{2k} \Bigg) \\
&= \frac{\lambda^p}{8\binom{n}{2}} \cdot \sum_i \left(2n\sum_{\ell\ge 0}\binom{2p}{2\ell}\Big(\frac{b_i\delta}{x_i}\Big)^{2\ell} + 2(n-2)\sum_{k\ge 0}\binom{p}{k}(-1)^k\Big(\frac{b_i\delta}{x_i}\Big)^{2k} \right) \\
&= \frac{\lambda^p}{4\binom{n}{2}} \cdot \sum_i \sum_{k\ge 0} \left(n\binom{2p}{2k} + (n-2)\binom{p}{k}(-1)^k \right) \left(\frac{b_i\delta}{x_i}\right)^{2k}.
\end{aligned}$$

Now fix $p = 1 - 1/n$. By Lemma 2.6.2, we have

$$\begin{aligned}
\mathbf{E}[\Lambda^{1-1/n}] &\le \frac{\lambda^{1-1/n}}{4\binom{n}{2}} \cdot \sum_i \sum_{k\ge 0} f_k(n, 1-1/n) \left(\frac{b_i\delta}{x_i}\right)^{2k} \\
&\le \frac{\lambda^{1-1/n}}{4\binom{n}{2}} \cdot \sum_i f_0(n, 1-1/n) \left(\frac{b_i\delta}{x_i}\right)^{0} \\
&= \frac{\lambda^{1-1/n}}{4\binom{n}{2}} \cdot n(n+(n-2)) = \lambda^{1-1/n}.
\end{aligned}$$

□

Lemma 2.6.4. *Let* $\mathbf{X}$ *be the output of* SRSimplify$(\mathbf{x}, \mathbf{a})$ *for some vectors* $\mathbf{x} \in (0,1)^n$, $\mathbf{a} \in (\mathbb{R}\setminus\{0\})^n$, *and* $n \ge 2$. *For any disjoint sets* $S, T \subseteq [n]$, *we have*

$$\mathbf{E}\left[\Big(\prod_{i\in S} X_i \prod_{i\in T}(1-X_i)\Big)^p\right] \le \Big(\prod_{i\in S} x_i \prod_{i\in T}(1-x_i)\Big)^p,$$

holds for $p = 1 - \frac{1}{n}$. *Furthermore, if all* $\{a_i\}_{i\in S}$ *are positive (or* $S = \emptyset$*) and all*

$\{a_i\}_{i\in T}$ are negative (or $T=\emptyset$), or vice versa, the inequality holds for $p=1$.

Proof. Let us define vectors $\mathbf{x}', \mathbf{a}'$ as

$$x'_i := \begin{cases} 1-x_i & i\in T \\ x_i & i\notin T \end{cases}, \qquad a'_i := \begin{cases} -a_i & i\in T \\ a_i & i\notin T \end{cases} \tag{2.10}$$

Let $\mathbf{X}' := \textsc{SRSimplify}(\mathbf{x}', \mathbf{a}')$ and $\mathbf{X}''$ be such that $X''_i = X_i$ if $i\notin T$ and $X''_i = 1 - X_i$ otherwise. It is not hard to verify that $\mathbf{X}'$ and $\mathbf{X}''$ have the same joint probability distribution. Then

$$\mathbf{E}\left[\Big(\prod_{i\in S} X_i \prod_{i\in T}(1-X_i)\Big)^p\right] = \mathbf{E}\left[\Big(\prod_{i\in S\cup T} X'_i\Big)^p\right] \le \Big(\prod_{i\in S\cup T} x'_i\Big)^p = \Big(\prod_{i\in S} x_i \prod_{i\in T}(1-x_i)\Big)^p$$

holds for $p = 1-\frac{1}{n}$. If all $\{a_i\}_{i\in S}$ are positive and all $\{a_i\}_{i\in T}$ are negative (or vice versa), then all $\{a'_i\}_{i\in S\cup T}$ have the same sign so this holds for $p=1$. $\square$

We are now ready to prove Theorem 2.6.1.

Proof of Theorem 2.6.1. Let $\lambda := \prod_{i\in S} x_i \prod_{i\in T}(1-x_i)$. Observe that the bound in Lemma 2.6.4 also holds for all values $p\in[0, 1-1/n]$ according to Jensen's inequality. By induction and using the bound in Lemma 2.6.4 with $p = 1-1/(t+1)$ on each iteration (which is valid as we have at least $t+1$ fractional values in the last step), we obtain

$$\mathbf{E}\left[\Big(\prod_{i\in S} X_i \prod_{i\in T}(1-X_i)\Big)^p\right] \le \lambda^p.$$

For n and t large enough, we have

$$\begin{aligned}\mathbf{E}\left[\prod_{i\in S} X_i \prod_{i\in T}(1-X_i)\right] &\le \mathbf{E}\left[\Big(\prod_{i\in S} X_i \prod_{i\in T}(1-X_i)\Big)^p\right] \\ &\le \lambda^{1-1/(t+1)} \approx \lambda\left(1-\frac{\log\lambda}{t}\right) \\ &\le \lambda\left(1-\frac{\log\alpha^m}{t}\right) = \lambda\left(1+\frac{m\log(1/\alpha)}{t}\right).\end{aligned}$$

□

Chapter 3: The k-median Problem

3.1 Problem definition

In this chapter, we study the classic k-median problem. In this problem, we are given a set $\mathcal{F}$ of facilities, a set $\mathcal{C}$ of clients, a budget k, and a symmetric distance-metric d on $\mathcal{F} \cup \mathcal{C}$. The goal is to open a subset of at most k facilities in $\mathcal{F}$ such that the total distance (or connection cost) from each client to its closest opened facility is minimized. That is, we want a subset $\mathcal{S} \subseteq \mathcal{F}$ of size at most k, which minimizes $\sum_{j \in \mathcal{C}} \min_{i \in S} d(i, j)$. Note that the only decision is which subset of the facilities to open.

This problem is known to be NP-hard, so there has been much work done on designing approximations with provable performance guarantees; indeed, virtually every major technique in approximation algorithms has been used and/or developed for this problem and its variants.

3.2 Prior work and Our contributions

Charikar, Guha, Tardos, and Shmoys used LP-rounding to achieve the first constant factor approximation ratio of $6\frac{2}{3}$ [33]. Then, Jain and Vazirani [25] applied Lagrangian Relaxation to remove the hard constraint of opening at most k facilities, effectively reducing the problem to an easier version known as the Uncapacitated Facility Location (UFL) problem. Using this technique together with primal-dual methods for UFL, they first find a bi-point solution, losing a factor of 3. They then round this bi-point solution to an integral feasible solution losing another multiplicative factor of 2, yielding a 6-approximation. Later, Jain, Mahdian, and Saberi (JMS) improved the approximation ratio of constructing the bi-point solution to 2, resulting in a 4-approximation [30]. Following this, a local-search-based $(3+\epsilon)$-approximation algorithm was developed by Arya et al. [34].

Recently, Li and Svensson's breakthrough work gave a $(1+\sqrt{3}+\epsilon)$-approximation algorithm for k-median [35]. To accomplish this, they defined an *α-pseudo-approximation* algorithm to be one that is an α-approximation which, however, opens $k + O(1)$ facilities, and showed – very surprisingly – how to use such an algorithm as a blackbox to construct a true $(\alpha + \epsilon)$-approximation algorithm. They then took advantage of this by giving a bi-point rounding algorithm which opens $k + O(1)$ facilities, but loses a factor of $\frac{1+\sqrt{3}}{2} + \epsilon$ instead of the previous 2. Together with the factor of 2 lost during the JMS bi-point construction algorithm, this yields the final approximation ratio. Letting N denote the input-size, the runtime of [35] is $N^{O(1/\epsilon^2)}$.

In this thesis, we give an improved bi-point rounding step to give an algorithm for k-median with improved approximation ratio and runtime. Section 4 presents an improved approximation for rounding bi-point solutions; we obtain $1.3371+\epsilon$ instead of $\frac{1+\sqrt{3}}{2} + \epsilon \sim 1.366 + \epsilon$. This yields a $2 \times 1.3371 + \epsilon \sim (2.675 + \epsilon)$-approximation algorithm for k-median, an improvement over Li and Svensson's $(2.733 + \epsilon)$. Using our dependent rounding technique in Chapter 2, we also improve dependence of the run-time on ϵ from $N^{O(1/\epsilon^2)}$ as in [35], to $N^{O((1/\epsilon)\log(1/\epsilon))}$.

3.3 An improved bi-point rounding algorithm

3.3.1 Preliminaries

Convex combinations of *two* integral solutions, with the corresponding convex combination of the number of open facilities being k, will be particularly useful for us.

Definition 3.3.1. *(Bi-point solution) Given a k-median instance $\mathcal{I}$, a bi-point solution is a pair $\mathcal{F}_1, \mathcal{F}_2 \subseteq \mathcal{F}$ such that $|\mathcal{F}_1| \leq k \leq |\mathcal{F}_2|$, along with reals $a, b \geq 0$ with $a + b = 1$ such that $a|\mathcal{F}_1| + b|\mathcal{F}_2| = k$. (That is, the convex combination of the two "solutions" is feasible for the natural LP relaxation of $\mathcal{I}$.) The cost of this bi-point solution is defined as $aD_1 + bD_2$, where D_1 and D_2 are the connection costs of $\mathcal{F}_1$ and $\mathcal{F}_2$ respectively.*

We refer to $\mathcal{F}_1, \mathcal{F}_2, a, b, D_1$, and D_2 as defined in Definition 3.3.1. As an initial goal (which we will relax shortly), suppose we are interested in an algorithm which rounds this bi-point solution to an integer solution of cost at most $\alpha(aD_1 + bD_2)$, for some α. As already mentioned, such an algorithm can be used to get a $(2 \times \alpha)$-approximation to k-median. Suppose a client j is closest to i_1 in solution $\mathcal{F}_1$, and i_2 in pseudo-solution[1] $\mathcal{F}_2$. *Ideally*, one would like to round the bi-point solution in such a way that i_1 is open with probability a, and i_2 is open with the remaining probability

[1]One that may open more than k facilities.

b. Then the expected connection cost of j would be exactly its contribution to the bi-point cost, and we would get a bi-point rounding factor of 1. The problem is that we cannot directly correlate this pair of events for every single client, while still opening only k facilities. Jain and Vazirani's approach is to pair each $i_1 \in \mathcal{F}_1$ with its closest neighbor in $\mathcal{F}_2$, and ensure that one of the two is open [25]. This approach loses at most a factor of 2, which is equal to the integrality gap of the k-median LP, and so is the best one might expect. However, Li and Svensson beat this factor by allowing their algorithm to open $k + c$ facilities. They then give a (surprising) method to convert such an algorithm to one that satisfies the budget-k constraint, adding ϵ to the approximation constant, and a factor of $n^{O(c/\epsilon)}$ to the runtime. This method actually runs the algorithm on a polynomial number of sub-instances of the original problem, and thus is not limited by the integrality gap of the original LP. Thus we obtain our relaxed goal:

Definition 3.3.2. *(Relaxed goal) Given a bi-point solution parametrized by $\mathcal{F}_1, \mathcal{F}_2, a, b, D_1$, and D_2 as in Definition 3.3.1, round it to an integer pseudo-solution using at most $k + f(\epsilon)$ facilities, and of cost at most $\alpha(aD_1 + bD_2)$ for $\alpha \sim 1.3371$. Here, $\epsilon > 0$ is an arbitrary constant.*

Definition 3.3.3 (Stars)**.** *For a given bi-point solution $a\mathcal{F}_1 + b\mathcal{F}_2$, we associate each facility $i_2 \in \mathcal{F}_2$ to its closest facility $i_1 \in \mathcal{F}_1$ (breaking ties arbitrarily). For each $i \in \mathcal{F}_1$, the set of i and its associated facilities in $\mathcal{F}_2$ is called a star. We refer to i as the center of the star and other facilities in the star as leaves. Also let $\mathcal{S}_i$ denote the set of leaves of the star with center i.*

Now we further partition the stars by their number of leaves. Let $\mathcal{T}_0$ be the set of stars with no leaves, $\mathcal{T}_1$ be the set of stars with one leaf, and $\mathcal{T}_2$ be the set of stars with at least 2 leaves. We call the stars in $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2$ as 0-stars, 1-stars, and 2-stars, respectively. Let $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ be the sets of centers of stars in $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2$, respectively. Let $\mathcal{L}_1, \mathcal{L}_2$ be the sets of leaves of stars in $\mathcal{T}_1, \mathcal{T}_2$, respectively. For a client j, let $i_1(j)$ and $i_2(j)$ denote the closest facilities to j in $\mathcal{F}_1$ and $\mathcal{F}_2$ respectively.

We also use the following notations: $\Delta_F := |\mathcal{F}_2| - |\mathcal{F}_1|, r_D := D_2/D_1, r_0 := |\mathcal{C}_0|/\Delta_F, r_1 := |\mathcal{C}_1|/\Delta_F$, $r_2 := |\mathcal{C}_2|/\Delta_F$, and $s_0 := 1/(1 + r_0)$. Note that if $\Delta_F = 0$, then $|\mathcal{F}_1| = |\mathcal{F}_2| = k$, and we may simply choose $\mathcal{F}_2$ as our solution, which has cost at most that of the bipoint solution. Thus we assume that $\Delta_F > 0$.

Now we describe a set of randomized algorithms to round a bi-point solution into a pseudo-solution which opens at most $k + O(1)$ facilities. In order to keep the number of extra facilities bounded, we consider several different cases depending on certain properties of the bi-point solution. In the main case, we get a $1.3371 + \epsilon$ approximation, utilizing WeightedDepRound2 to open only $O(\log(1/\epsilon))$ extra facilities. In the edge cases, we are able to use weaker, but simpler techniques to obtain the same bound.

3.3.2 Main case: $s_0 \geq 5/6, b \in [0.508, 3/4], r_D \in [19/40, 2/3]$, and $r_1 > 1$

For each 1-star with center i and leaf i', we define the following ratio. (Note that $\Delta_F > 0$ implies $\mathcal{L}_2$ is nonempty.)

$$g_i = \frac{d(i,i')}{\min_{j \in \mathcal{L}_2} d(i,j)}.$$

We partition the set $\mathcal{T}_1$ into sets $\mathcal{T}_{1A}$ of *long* stars and $\mathcal{T}_{1B}$ of *short* stars as follows. We sort all the stars in $\mathcal{T}_1$ in decreasing order of g_i. Let $\mathcal{T}_{1A}$ be the set of the first $\lceil a\Delta_F \rceil$ stars of $\mathcal{T}_1$ and $\mathcal{T}_{1B} := \mathcal{T}_1 \setminus \mathcal{T}_{1A}$. Also let $\mathcal{C}_{1A}$ and $\mathcal{C}_{1B}$ be the sets of centers of stars in $\mathcal{T}_{1A}$ and $\mathcal{T}_{1B}$, respectively. Similarly, let $\mathcal{L}_{1A}$ and $\mathcal{L}_{1B}$ be the corresponding sets of leaves. Note that $\mathcal{T}_{1A}$ is well-defined since $|\mathcal{T}_1|/\Delta_F = r_1 > 1$ implies $|\mathcal{T}_1| > \Delta_F$.

Next, we describe a rounding scheme called $\mathcal{A}(p_0, p_{1A}, q_{1A}, p_{1B}, q_{1B}, p_2, q_2)$ which is the main procedure of our algorithm. The purpose of $\mathcal{A}$ is to (for $X \in \{0, 1_A, 1_B, 2\}$) randomly open roughly p_X fraction of facilities in $\mathcal{C}_X$, and q_X fraction of facilities in $\mathcal{L}_X$, while maintaining the important property that if any leaves of a star are closed, its center will be opened – *except* in some cases where we completely close all stars in $\mathcal{T}_{1A}$.

When $p_2 \neq 0$, we further partition $\mathcal{T}_2$ into "large" and "small" stars (as in [35]). For a given parameter $\eta > 0$, we say that a star centered at $i \in \mathcal{C}_2$ is large if $|\mathcal{S}_i| \geq 1/(p_2\eta)$ and small otherwise. Let $\beta = \min\{q_2, 1-q_2\}$ and $c = \lceil \frac{16}{3\beta^2} \rceil$. Then, we group the small stars according to their sizes: For each $s = 1, \dots, \lceil \log_{1+\beta}(1/(p_2\eta)) \rceil - 1$, let $\mathcal{G}_s := \{i \in \mathcal{C}_2 : (1+\beta)^s \le |\mathcal{S}_i| < (1+\beta)^{s+1}\}$.

3.3.2.1 Main algorithm

Below we define Algorithm $\mathcal{A}$ and its subroutine ROUND2STARS. The main algorithm will simply run $\mathcal{A}$ with 9 different sets of parameters and return the solution with minimum connection cost. We refer to these calls of $\mathcal{A}$ as algorithms $\mathcal{A}_1, \cdots, \mathcal{A}_9$. See Table 3.1 for a complete set of parameters. It is easy to see that all numbers in the table belong to $[0, 1]$ as $b \ge a$, $0 \le s_0 \le 1$, and $r_2 \ge 0$.

Algorithm 8 $\mathcal{A}(p_0, p_{1A}, q_{1A}, p_{1B}, q_{1B}, p_2, q_2)$

```
1: Randomly open a subset of size $\lceil p_0|\mathcal{C}_0| \rceil$ of $\mathcal{C}_0$.
2: Take a random permutation of $\mathcal{T}_{1A}$. Open the centers of the first $\lceil p_{1A}|\mathcal{T}_{1A}| \rceil$
   stars and the leaves of the last $\lceil q_{1A}|\mathcal{T}_{1A}| \rceil$.
3: Take a random permutation of $\mathcal{T}_{1B}$. Open the centers of the first $\lceil p_{1B}|\mathcal{T}_{1B}| \rceil$
   stars and the leaves of the last $\lceil q_{1B}|\mathcal{T}_{1B}| \rceil$.
4: if $p_2 = 1$ or $p_2 = 0$ then
5:     Open all or none of $\mathcal{C}_2$, respectively. Also open a random subset of size $\lceil q_2|\mathcal{L}_2| \rceil$
       of $\mathcal{L}_2$.
6: else
7:     ROUND2STARS $(p_2, q_2)$.
8: Return the set of all opened facilities.
```

Algorithms	p_0	p_{1A}	q_{1A}	p_{1B}	q_{1B}	p_2	q_2
$\mathcal{A}_1$	0	0	1	0	1	as_0	$1 - as_0$
$\mathcal{A}_2$	1	0	1	0	1	$1 - bs_0$	bs_0
$\mathcal{A}_3$	1	0	1	1	0	$1 - bs_0$	bs_0
$\mathcal{A}_4$	1	1	0	0	1	$1 - bs_0$	bs_0
$\mathcal{A}_5$	1	1	0	1	0	$1 - bs_0$	bs_0
$\mathcal{A}_6$	1	1	1	1	0	$1 - (b-a)s_0$	$(b-a)s_0$
$\mathcal{A}_7$	1	1	0	1	0	1	$\frac{1}{2}bs_0$
$\mathcal{A}_8$	0	0	0	0	1	0	1
$\mathcal{A}_9$	a	a	b	a	b	a	b

Table 3.1: The main algorithm makes 9 calls to $\mathcal{A}$, with the above parameters.

Algorithm 9 ROUND2STARS(p_2, q_2)

```
1: Open the centers of all large stars. Let $\mathcal{C}'_2$ be the set of these centers, and let $\mathcal{L}'_2$ be the set of their leaves. Randomly open a subset of size $\lceil q_2(|\mathcal{L}'_2| - |\mathcal{C}'_2|)\rceil$ of $\mathcal{L}'_2$.
2: for $s = 1, \ldots, \lceil \log_{1+\beta}(1/(p_2\eta))\rceil - 1$ do
3:     Let $A, P$ be vectors with $A_i = |\mathcal{S}_i| - 1$ and $P_i = q_2$ for $i \in \mathcal{G}_s$.
4:     Let $X$ be the vector returned by WEIGHTEDDEPROUND2 on $A$ and $P$.
5:     For all integer elements $X_i$, if $X_i = 1$, open all facilities in $\mathcal{S}_i$. Else, open the center $i$.
6:     Let $X_{i^*}$ be the fractional element (if any). Open the center of $\mathcal{S}_{i^*}$ and a random set of size $\lceil X_{i^*}|\mathcal{S}_{i^*}|\rceil$ of $\mathcal{S}_{i^*}$.
7:     Pick $\min\{c, |\mathcal{G}_s|\}$ centers of stars in $\mathcal{G}_s$ uniformly at random and open them if not already opened.
```

The algorithm itself runs in linear time. However, when we use Li and Svensson's algorithm to convert our pseudo-solution to a feasible one, it will take time $O(n^{O(C/\epsilon)})$ in total, where C is the number of extra facilities we open. So it is important that C is a (preferably small) constant. A few of these extra facilities come from handling basic rounding (e.g. $\lceil q_2|\mathcal{L}_2|\rceil$), however, the majority come from handling the positive correlation within the groups $\mathcal{G}_s$. Li and Svensson considered $O(1/\eta)$ groups of stars, each with uniform size, bounded the positive correlation by adding a few extra facilities per group, and showed that the total cost is only blown up by a factor of $(1 + \eta)$. The near-independence property gives a bound on the positive correlation, so that we may compensate for it by adding $O(1/\beta^2)$ extra facilities per group. Thus, β must be bounded away from zero, which strongly motivates our restriction of the domain of the main algorithm.

Note that if we run $\mathcal{A}$ with parameters $p_0 = p_{1A} = p_{1B} = p_2 = a$, and $q_{1A} = q_{1B} = q_2 = b$, the resulting algorithm is essentially the same as that given in [35]. (The set of algorithms we use subsumes the need for this one.) The main difference in this case is that we need to open only $O(\log(1/\epsilon))$ extra facilities instead of $O(1/\epsilon)$.

3.3.2.2 Bounding the number of opened facilities

Since our main algorithm will return one of the solutions by $\mathcal{A}_1, \cdots, \mathcal{A}_9$, we need to show that none of these will open too many facilities. Algorithm 9 essentially partitions all stars into a constant number of groups. Consider the *budget* of each group, which is the expected number of facilities opened in that group if we independently open each facility in $\mathcal{C}_X$ with probability p_X and each in $\mathcal{L}_X$ with q_X. We want to show that for each group, the number of facilities opened is always within an additive constant of that group's budget. The trickiest groups are the groups of small stars $\{\mathcal{G}_s\}$.

Lemma 3.3.1. *For each group $\mathcal{G}_s$, let $\mathcal{C}(s)$ and $\mathcal{L}(s)$ be the set of centers and leaves of stars in $\mathcal{G}_s$ respectively. Then* Round2Stars *always opens at most $p_2|\mathcal{C}(s)| + q_2|\mathcal{L}(s)| + c + 2$ facilities in $\mathcal{C}(s) \cup \mathcal{L}(s)$.*

Proof. Since WeightedDepRound2 preserves the weighted sum, we have with probability 1 that

$$\sum_{i \in \mathcal{C}(s)} X_i(|\mathcal{S}_i| - 1) = \sum_{i \in \mathcal{C}(s)} q_2(|\mathcal{S}_i| - 1).$$

The number of facilities opened in lines 5 and 6 is at most

$$
\begin{aligned}
&\sum_{i \in \mathcal{C}(s), i \neq i^*} \big(X_i|\mathcal{S}_i| + (1 - X_i)\big) + 1 + \lceil X_{i^*}|\mathcal{S}_{i^*}|\rceil \\
&\leq \sum_{i \in \mathcal{C}(s), i \neq i^*} X_i(|\mathcal{S}_i| - 1) + (|\mathcal{C}(s)| - 1) + 2 + X_{i^*}(|\mathcal{S}_{i^*}| - 1) + X_{i^*} \\
&= \sum_{i \in \mathcal{C}(s)} X_i(|\mathcal{S}_i| - 1) + (|\mathcal{C}(s)| - 1) + 2 + X_{i^*} \\
&= \sum_{i \in \mathcal{C}(s)} q_2(|\mathcal{S}_i| - 1) + |\mathcal{C}(s)| + 1 + X_{i^*} \\
&\leq \sum_{i \in \mathcal{C}(s)} q_2(|\mathcal{S}_i| - 1) + |\mathcal{C}(s)| + 2 \\
&= \sum_{i \in \mathcal{C}(s)} (q_2(|\mathcal{S}_i| - 1) + 1) + 2 = \sum_{i \in \mathcal{C}(s)} (q_2|\mathcal{S}_i| + p_2) + 2 = p_2|\mathcal{C}(s)| + q_2|\mathcal{L}(s)| + 2,
\end{aligned}
$$

where in the penultimate step we have used that $p_2 + q_2 = 1$ whenever Round2Stars is called. (This follows from $\mathcal{A}_1 \ldots \mathcal{A}_9$, except $\mathcal{A}_7$ where Round2Stars would never be called.) The lemma follows because we open at most c additional facilities in line 7. □

Note that the number of groups of small stars is at most $\log_{1+\beta}(1/(p_2\eta))$, and we open at most $c + 2 = \lceil 16/(3\beta^2)\rceil + 2$ additional facilities in each group. It is straightforward to see that the other groups ($\mathcal{T}_{1A}$, $\mathcal{T}_{1B}$, and large stars) only open a constant number of extra facilities, and so our total budget is violated by only a constant amount. The following claim shows that β and p_2 are strictly greater than zero (i.e., c and the number of groups are upper-bounded by real constants.)

Claim 3.3.1. *When* Round2Stars *is called, we have $\beta > 1/75$ and $p_2 \geq 5/24$.*

Proof. ROUND2STARS is only called during $\mathcal{A}_1, \ldots, \mathcal{A}_6$. (In A_7 and A_8, line 5 is called instead.) Consider possible values of p_2 and q_2 in Table 3.1. Recall that $s_0 \in [5/6, 1]$, $b \in [0.508, 3/4]$ and $a + b = 1$. The minimum of $\beta = \min\{q_2, 1 - q_2\}$ is attained in $\mathcal{A}_6$ when $b = 0.508$ and $s_0 = 5/6$; here $q_2 = (b - a)s_0 = (0.508 - 0.492) \cdot 5/6 = 1/75$. Also, the minimum of p_2 is attained at $p_2 = as_0$, $a = 1/4$, and $s_0 = 5/6$. □

Since we open basically $O(\frac{1}{\beta^3} \log(\frac{1}{\eta}))$ extra facilities, these small lower bounds lead to poor constants. Significant improvement may be made by further splitting the cases, and carefully choosing the set of algorithms used in each. However, in order to avoid further complicating the algorithm and its analysis, we do not attempt to optimize these values here.

Lemma 3.3.2. *For any given set of parameters* $\{p_0, p_{1A}, q_{1A}, p_{1B}, q_{1B}, p_2, q_2\}$ *in Table 3.1, $\mathcal{A}$ will open at most $E + O(\log(1/\eta))$ facilities with probability* 1*, where*

$$E := p_0|\mathcal{C}_0| + p_{1A}|\mathcal{C}_{1A}| + q_{1A}|\mathcal{C}_{1A}| + p_{1B}|\mathcal{C}_{1B}| + q_{1B}|\mathcal{C}_{1B}| + p_2|\mathcal{C}_2| + q_2|\mathcal{L}_2|.$$

Proof. We consider Algorithm 17. It is easy to see that

- In line 1, we open at most $p_0|\mathcal{C}_0| + 1$ facilities,
- In line 2, we open at most $p_{1A}|\mathcal{C}_{1A}| + q_{1A}|\mathcal{C}_{1A}| + 2$ facilities,
- In line 3, we open at most $p_{1B}|\mathcal{C}_{1B}| + q_{1B}|\mathcal{C}_{1B}| + 2$ facilities,
- If line 5 is executed then we open at most $p_2|\mathcal{C}_2| + q_2|\mathcal{L}_2| + 1$ facilities,

- Otherwise, Round2Stars is called:
  - In line 1, the number of opened facilities is

$$|\mathcal{C}_2'| + \lceil q_2(|\mathcal{L}_2'| - |\mathcal{C}_2'|) \rceil \leq 1 + |\mathcal{C}_2'| + q_2(|\mathcal{L}_2'| - |\mathcal{C}_2'|) = p_2|\mathcal{C}_2'| + q_2|\mathcal{L}_2'| + 1,$$

    where the equality follows due to the fact that $1 - q_2 = p_2$ whenever Round2Stars is called.
  - By Lemma 3.3.1 and Claim 3.3.1, the number of facilities opened by the for loop (lines $3 \dots 7$) is at most

$$\begin{aligned} & \sum_{s=1}^{\lceil \log_{1+\beta}(1/(p_2\eta)) \rceil - 1} (p_2|\mathcal{C}(s)| + q_2|\mathcal{L}(s)| + c + 2) \\ = & \sum_{s=1}^{\lceil \log_{1+\beta}(1/(p_2\eta)) \rceil - 1} (p_2|\mathcal{C}(s)| + q_2|\mathcal{L}(s)|) + O(\log(1/\eta)). \end{aligned}$$

The lemma follows by taking the sum of opened facilities in each case.

□

The $O(\log(1/\eta))$ term comes as a result of us opening $O(\log(1/\eta))$ small groups $\mathcal{G}_s$. The parameters in $\mathcal{A}_1, \cdots, \mathcal{A}_8$ are carefully chosen so that the total budget $E \approx k$ in each case. This gives us the following result.

Lemma 3.3.3. *Algorithms $\mathcal{A}_1, \cdots, \mathcal{A}_9$ will always open at most $k + O(\log(1/\eta))$ facilities.*

Proof. Since $b \in [1/2, 3/4]$ and $s_0 \leq 1$, p_2 is bounded away from 0 in $\mathcal{A}_1, \dots, \mathcal{A}_9$. Note that Round2Stars is not called in $\mathcal{A}_7$ and $\mathcal{A}_8$; at most $E + 1$ facilities can be opened in these two algorithms. By Lemma 3.3.2, it suffices to show that $E \le k+1$. The proof is straightforward. We substitute the parameters in Table 3.1 and $s_0 = \frac{1}{1+|\mathcal{C}_0|/\Delta_F} = 1 + \frac{|\mathcal{C}_0|}{|\mathcal{C}_2|-|\mathcal{L}_2|}$ to compute E in each case. We use simple facts such as $|\mathcal{C}_{1A}| = |\mathcal{L}_{1A}|$, $|\mathcal{C}_{1B}| = |\mathcal{L}_{1B}|$, and $|\mathcal{C}_{1A}| + |\mathcal{C}_{1B}| = |\mathcal{C}_1|$ to further simplify the expression. Also recall that $|\mathcal{C}_{1A}| = \lceil a\Delta_F \rceil$, and thus $a\Delta_F \le |\mathcal{C}_{1A}| \le a\Delta_F + 1$. By definition, we have $\Delta_F = |\mathcal{L}_2| - |\mathcal{C}_0| - |\mathcal{C}_2|$ and $2|\mathcal{C}_2| \le |\mathcal{L}_2|$.

- For $\mathcal{A}_1$, we have

$$\begin{aligned} E &= |\mathcal{C}_{1A}| + |\mathcal{C}_{1B}| + as_0|\mathcal{C}_2| + (1-as_0)|\mathcal{L}_2| \\ &= |\mathcal{C}_1| + a|\mathcal{C}_0| + a|\mathcal{C}_2| + b|\mathcal{L}_2| \\ &= a(|\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2|) + b(|\mathcal{C}_1| + |\mathcal{L}_2|) = a|\mathcal{F}_1| + b|\mathcal{F}_2| = k. \end{aligned}$$

- For $\mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4$, and $\mathcal{A}_5$, substituting the parameters gives the same E:

$$\begin{aligned} E &= |\mathcal{C}_0| + |\mathcal{C}_1| + (1-bs_0)|\mathcal{C}_2| + bs_0|\mathcal{L}_2| \\ &= |\mathcal{C}_0| - b|\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2| - b|\mathcal{C}_2| + b|\mathcal{L}_2| \\ &= a|\mathcal{F}_1| + b|\mathcal{F}_2| = k. \end{aligned}$$

- For $\mathcal{A}_6$, we have

$$\begin{aligned}
E &= |\mathcal{C}_0| + |\mathcal{C}_{1A}| + |\mathcal{C}_{1A}| + |\mathcal{C}_{1B}| + (1-(b-a)s_0)|\mathcal{C}_2| + (b-a)s_0|\mathcal{L}_2| \\
&= |\mathcal{C}_0| + |\mathcal{C}_{1A}| + |\mathcal{C}_1| + (1-(b-a)s_0)|\mathcal{C}_2| + (b-a)s_0|\mathcal{L}_2| \\
&\leq 1 + |\mathcal{C}_0| + a\Delta_F + |\mathcal{C}_1| + (1-(b-a)s_0)|\mathcal{C}_2| + (b-a)s_0|\mathcal{L}_2| \\
&= 1 + |\mathcal{C}_0| - b|\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2| - b|\mathcal{C}_2| + b|\mathcal{L}_2| \\
&= 1 + a|\mathcal{F}_1| + b|\mathcal{F}_2| = k+1.
\end{aligned}$$

- For $\mathcal{A}_7$, we have

$$\begin{aligned}
E &= |\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2| + \frac{b}{2(1/s_0)}|\mathcal{L}_2| \\
&\leq |\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2| + \frac{b}{1/s_0 + r_2}|\mathcal{L}_2| \\
&= |\mathcal{C}_0| - b|\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2| - b|\mathcal{C}_2| + b|\mathcal{L}_2| \\
&= a|\mathcal{F}_1| + b|\mathcal{F}_2| = k.
\end{aligned}$$

- For $\mathcal{A}_8$, we have

$$\begin{aligned} E &= |\mathcal{C}_{1B}| + |\mathcal{L}_2| \\ &= |\mathcal{C}_1| - |\mathcal{C}_{1A}| + |\mathcal{L}_2| \\ &\leq |\mathcal{C}_1| - a\Delta_F + |\mathcal{L}_2| \\ &\leq |\mathcal{C}_1| - \Delta_F + b\Delta_F + |\mathcal{L}_2| \\ &= |\mathcal{F}_1| + b\Delta_F = k. \end{aligned}$$

- For $\mathcal{A}_9$ (this is exactly Li-Svensson algorithm), we have

$$\begin{aligned} E &= a|\mathcal{C}_0| + a|\mathcal{C}_1| + b|\mathcal{L}_1| + a|\mathcal{C}_2| + b|\mathcal{L}_2| \\ &= a|\mathcal{F}_1| + b|\mathcal{F}_2| = k. \end{aligned}$$

□

3.3.2.3 Cost analysis

We now derive bounds for the expected connection cost of a single client. For each client $j \in \mathcal{C}$, let $i_1(j)$ and $i_2(j)$ be the client's closest facilities in $\mathcal{F}_1$ and $\mathcal{F}_2$, and let $d_1(j)$ and $d_2(j)$ be their respective distances from j. Also let $i_3(j)$ be the center of the star containing $i_2(j)$. (Where obvious, we omit the parameter j.) We will obtain several different upper bounds, depending on the class of the star in which $i_1(j)$ and $i_2(j)$ lie.

A key characteristic of Algorithm 17 is that for any star in class $Y \in \{1_A, 1_B, 2\}$, as long as $p_Y + q_Y \geq 1$, it will always open either the star's center or all of the star's leaves. By definition of stars, we know i_3 is not too far away. We will slightly abuse notation and let i and $\bar{i}$ represent the events that facility i is opened or closed, respectively. By considering these probabilities, we obtain the following two bounds, similar to the one used in [35]. See Figure 4.4.

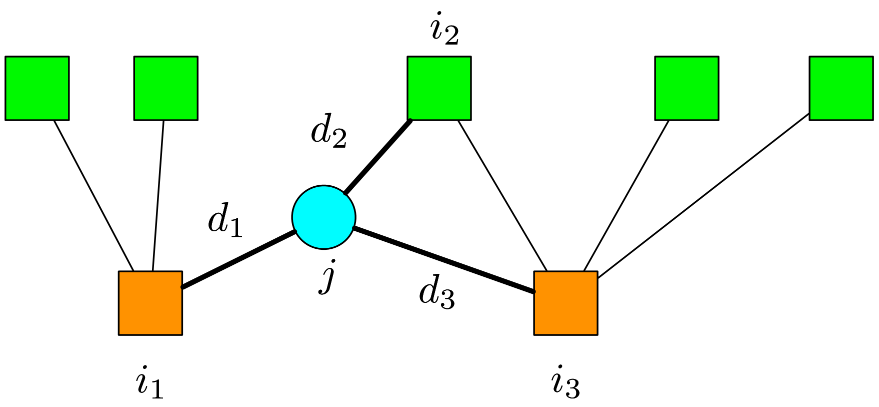

Figure 3.1: Illustration of i_1, i_2, and i_3. Observe that $d_3 \leq d_2 + d(i_2, i_3) \leq d_2 + d(i_1, i_2) \leq d_1 + 2d_2$.

Lemma 3.3.4. *Let j be a client. Suppose we are running one of algorithms $\mathcal{A}_1$ to $\mathcal{A}_7$,* OR *we are running $\mathcal{A}_8$* and *$i_2(j) \notin \mathcal{L}_{1A}$. Then the expected connection cost of j after running Algorithm 17 is bounded above by both $c_{213}(j) := d_2 + \Pr[\bar{i}_2](d_1 - d_2) + 2\Pr[\bar{i}_1\bar{i}_2]d_2$ and $c_{123}(j) := d_1 + \Pr[\bar{i}_1](d_2 - d_1) + \Pr[\bar{i}_1\bar{i}_2](d_1 + d_2)$.*

Proof. For all these clients, we know that at least one of i_2 or i_3 will always be open. First consider the case that $i_1 \neq i_3$. Then the facilities are as shown below. Observe by the construction of stars, i_2 cannot be closer to i_1 than i_3 (otherwise i_1 would be its center). Thus $d(i_2, i_3) \leq d(i_2, i_1) \leq d_1 + d_2$. It follows by the triangle inequality that $d(j, i_3) \leq d(j, i_2) + d(i_2, i_3) \leq d_1 + 2d_2$.

Now let us connect j to i_2 if open. Else, connect to i_1 if open. Else, connect to i_3. The actual facility which j connects to can only be closer than any of these. Thus, this yields the following upper bound for the expected connection cost of j.

$$\begin{aligned} c_{213}(j) &:= \Pr[i_2]d_2 + Pr[i_1\bar{i}_2]d_1 + \Pr[\bar{i}_1\bar{i}_2](d_1 + 2d_2) \\ &= \Pr[i_2]d_2 + \Pr[\bar{i}_2]d_1 + 2\Pr[\bar{i}_1\bar{i}_2]d_2 \\ &= d_2 + \Pr[\bar{i}_2](d_1 - d_2) + 2\Pr[\bar{i}_1\bar{i}_2]d_2, \end{aligned} \tag{3.1}$$

where the subscript corresponds to the order in which we try connecting to facilities. Alternatively, we may connect j first to i_1 if open, else i_2 if open, else i_3. This gives the equally valid bound

$$\begin{aligned} c_{123}(j) &:= \Pr[i_1]d_1 + \Pr[\bar{i}_1 i_2]d_2 + \Pr[\bar{i}_1\bar{i}_2](d_1 + 2d_2) \\ &= \Pr[i_1]d_1 + \Pr[\bar{i}_1]d_2 + \Pr[\bar{i}_1\bar{i}_2](d_1 + d_2) \\ &= d_1 + \Pr[\bar{i}_1](d_2 - d_1) + \Pr[\bar{i}_1\bar{i}_2](d_1 + d_2). \end{aligned} \tag{3.2}$$

Now consider the remaining case that $i_1 = i_3$. In this case, at least one of i_1 or i_2 will always be open. Again, depending on which facility we attempt to connect to first, we can obtain either of two bounds:

$$\begin{aligned} c_{21}(j) &:= \Pr[i_2]d_2 + \Pr[\bar{i}_2]d_1 \leq c_{213}(j) \\ c_{12}(j) &:= \Pr[i_1]d_1 + \Pr[\bar{i}_1]d_2 \leq c_{123}(j). \end{aligned}$$

Thus (3.1) and (3.2) are valid bounds in both cases. □

In $\mathcal{A}_8$, $p_{1A} = q_{1A} = 0$, meaning all stars in $\mathcal{T}_{1A}$ have both center and leaf closed, so if $i_2 \in \mathcal{L}_{1A}$ the previous bound does not hold. In this case, let i_4 be the closest leaf of a 2-star to i_3. Recall the definition of g_i; this gives us information on the distance to i_4. Let $g := \min_{i \in \mathcal{C}_{1A}} g_i$ be the minimum value over all stars in $\mathcal{T}_{1A}$. Then we may bound the cost to i_4 (or its center, in the worst case) as follows:

Lemma 3.3.5. *Let j be a client such that $i_2(j) \in \mathcal{L}_{1A}$. Then the expected connection cost of j, when running A_8, is bounded above by $c_{145}(j) := d_1 + \Pr[\bar{i}_1]\left(2d_2 + \frac{1}{g}(d_1 + d_2)\right) + \Pr[\bar{i}_1\bar{i}_4]\frac{1}{g}(d_1 + d_2)$.*

Proof. For these clients it is possible that i_1, i_2 and i_3 are all closed. Let i_4 be the closest facility in $\mathcal{L}_2$ to i_3, and let i_5 be the center of i_4. Then by definition, we have

$$g_{i_3} = \frac{d(i_3, i_2)}{d(i_3, i_4)}.$$

This yields the following bound on $d(i_3, i_4)$ (and thus $d(i_4, i_5)$):

$$d(i_4, i_5) \leq d(i_3, i_4) = \frac{1}{g_{i_3}} d(i_2, i_3) \leq \frac{1}{g}(d_1 + d_2).$$

Now we know that if i_4 is closed, then i_5 must be open. We also know that i_2 and i_3 will always be closed. In the case that $i_1 \neq i_3$ (which is shown above), we will try connecting, in order, to i_1, i_4, and i_5, connecting to the first one which is open. This yields the following bound:

$$\begin{aligned}c_{145}(j) :=& \Pr[i_1]d_1 + \Pr[\bar{i}_1 i_4]\left(d_1 + 2d_2 + \frac{1}{g}(d_1+d_2)\right) + \Pr[\bar{i}_1\bar{i}_4]\left(d_1 + 2d_2 + \frac{2}{g}(d_1+d_2)\right) \\ =& d_1 + \Pr[\bar{i}_1]\left(2d_2 + \frac{1}{g}(d_1+d_2)\right) + \Pr[\bar{i}_1\bar{i}_4]\frac{1}{g}(d_1+d_2). \qquad (3.3)\end{aligned}$$

For the case that $i_1 = i_3$, we have the below situation: Here i_1 and i_2 are always closed, so we try connecting first to i_4, then to i_5, giving the following bound:

$$c_{45}(j) := d_1 + \frac{1}{g}(d_1+d_2) + \Pr[\bar{i}_4]\frac{1}{g}(d_1+d_2).$$

In this case $\Pr[\bar{i}_1] = 1$. Also since $i_1 \in \mathcal{C}_{1A}$ and $i_4 \in \mathcal{L}_2$ are in different types of stars, they are rounded independently, so $\Pr[\bar{i}_1\bar{i}_4] = \Pr[\bar{i}_1]\Pr[\bar{i}_4] = \Pr[\bar{i}_4]$. Thus, $c_{45}(j) \le c_{145}(j)$, and the claim still holds. □

These two lemmas provide a valid bound for all clients. However, the bound in Lemma 3.3.5 may be very poor if g is small. To balance this, we provide another bound which does well for small g.

Lemma 3.3.6. *Let j be a client such that $i_1(j) \in \mathcal{C}_{1B}$ and $i_2(j) \in \mathcal{L}_2$. Then in all algorithms, the expected cost of j is bounded above by both of the following:*

$$\begin{aligned}c_{210}(j) &:= d_2 + \Pr[\bar{i}_2](d_1 - d_2) + \Pr[\bar{i}_1\bar{i}_2]g(d_1+d_2), \\ c_{120}(j) &:= d_1 + \Pr[\bar{i}_1](d_2 - d_1) + \Pr[\bar{i}_1\bar{i}_2](d_1 - d_2 + g(d_1+d_2)).\end{aligned}$$

Proof. In this case $i_1 \in \mathcal{C}_{1B}$. Let i_0 be the leaf attached to i_1. Again, we know that if i_0 is closed, i_1 will be open. Recall that by definition $g_i = \frac{d(i,i')}{\min_{j \in \mathcal{L}_2} d(i,j)}$, where i and i' are the center and leaf, respectively of a 1-star. Applying this to i_1 and i_0, we have

$$d(i_1, i_0) = g_{i_1} \min_{i \in \mathcal{L}_2} d(i_1, i) \leq g \cdot d(i_1, i_2) \leq g(d_1 + d_2).$$

Now we may try connecting in order i_2, i_1, i_0, or alternatively, in order i_1, i_2, i_0, yielding the following bounds:

$$\begin{aligned} c_{210}(j) &:= \Pr[i_2]d_2 + Pr[i_1\bar{i}_2]d_1 + \Pr[\bar{i}_1\bar{i}_2](d_1 + g(d_1 + d_2)) \\ &= d_2 + \Pr[\bar{i}_2](d_1 - d_2) + Pr[\bar{i}_1\bar{i}_2]g(d_1 + d_2) \qquad (3.4) \\ c_{120}(j) &:= \Pr[i_1]d_1 + \Pr[\bar{i}_1 i_2]d_2 + \Pr[\bar{i}_1\bar{i}_2](d_1 + g(d_1 + d_2)) \\ &= d_1 + \Pr[\bar{i}_1](d_2 - d_1) + \Pr[\bar{i}_1\bar{i}_2](d_1 - d_2 + g(d_1 + d_2)). \qquad (3.5) \end{aligned}$$

Note that by definition of $i_2(j)$, we can say $d_2 = d(j, i_2) \leq d(j, i_0) \leq d_1 + g(d_1 + d_2)$, which implies $(d_1 - d_2 + g(d_1 + d_2)) \geq 0$, a fact that will be used later. $\square$

(Note: as we observe in the proof of the above, the coefficient $(d_1 - d_2 + g(d_1 + d_2))$ is non-negative.)

The following lemma relates the probabilities in the above bounds to the parameters of the algorithm. In particular, we take advantage of properties of WeightedDepRound2 as described in Chapter 2.

Lemma 3.3.7. *Let i_1 and i_2 be any two facilities in $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Let $X, Y \in \{0, 1_A, 1_B, 2\}$ be the classes such that $i_1 \in \mathcal{C}_X$ and $i_2 \in \mathcal{L}_Y$. Then for any*

$\mathcal{A}(p_0, p_{1A}, q_{1A}, p_{1B}, q_{1B}, p_2, q_2)$ *in Table 3.1, the following are true:*

$$\Pr[\bar{i}_1] \leq 1 - p_X, \tag{3.6}$$

$$\Pr[\bar{i}_2] \leq (1+\eta)(1 - q_Y), \tag{3.7}$$

$$\Pr[\bar{i}_1 \bar{i}_2] \leq (1+\eta)(1 - p_X)(1 - q_Y). \tag{3.8}$$

Proof. Consider i_1. Suppose $i_1 \in \mathcal{C}_X$. If $X \in \{0, 1_A, 1_B\}$, we have $\Pr[i_1] \geq p_X$ (by lines 1, 2, and 3 of Algorithm 17). Otherwise $X = 2$. If $p_2 = 0$ or $p_2 = 1$, line 5 of Algorithm 17 is executed and $\Pr[i_1] = p_2$ exactly. Else, we run ROUND2STARS. If i_1 is part of a large star, then it is always opened so $\Pr[\bar{i}_1] = 0$. Else, i_1 is in a small star, and we have $\Pr[\bar{i}_1] \leq \Pr[X_{i_1} = 1] \leq \mathbf{E}[X_{i_1}] = q_2 = 1 - p_2$. This holds because $\bar{i}_1$ only occurs when $X_{i_1} = 1$. In all cases (3.6) holds.

Consider i_2. Suppose $i_2 \in \mathcal{L}_Y$. If $Y \in \{1_A, 1_B\}$, we have $\Pr[i_2] \geq q_Y$. Otherwise $Y = 2$. Again, if line 5 of Algorithm 17 is executed, $\Pr[i_2] \geq q_2$. Else, we run ROUND2STARS. Consider the case that $i_2 \in \mathcal{L}'_2$ is part of a large star. Recall that large stars have at least $1/(p_2\eta) = 1/((1-q_2)\eta)$ leaves. Then

$$\Pr[i_2] \geq \frac{q_2(|\mathcal{L}'_2| - |\mathcal{C}'_2|)}{|\mathcal{L}'_2|} \geq q_2 - \frac{|\mathcal{C}'_2|}{|\mathcal{L}'_2|} \geq q_2 - (1-q_2)\eta = 1 - (1-q_2)(1+\eta).$$

Otherwise, i_2 is part of some small star, with center i_3. If X_{i_3} is 1 or 0, by line 5, $\Pr[i_2] = X_{i_3}$. If $0 < X_{i_3} < 1$, then by line 6, $\Pr[i_2] \geq X_{i_3}$. So in any case, we have $\Pr[i_2 | X_{i_3} = x] \geq x$. Note that each indicator returned by WEIGHTEDDEPROUND2 can only take finitely many values in $[0,1]$. Letting $\mathcal{U}$ be the set of these values, we have

$$\Pr[i_2] = \sum_{x \in \mathcal{U}} \Pr[i_2 | X_{i_3} = x] \Pr[X_{i_3} = x] \geq \sum_{x \in \mathcal{U}} x \Pr[X_{i_3} = x] = \mathbf{E}[X_{i_3}] = q_2.$$

In all cases (3.7) holds.

Now consider both i_1 and i_2. There are many cases to consider, but most of them are easy. If i_1 and i_2 belong to stars of different classes, then they are opened independently, so $\Pr[\bar{i}_1\bar{i}_2] = \Pr[\bar{i}_1]\Pr[\bar{i}_2] \leq (1+\eta)(1-p_X)(1-q_Y)$. For the remaining cases, $i_1 \in \mathcal{C}_X$ and $i_2 \in \mathcal{L}_X$ for the same class $X \in \{1_A, 1_B, 2\}$. There is a special case where $X = 1_A$, and we are running $\mathcal{A}_8$. In this case, $p_{1A} = q_{1A} = 0$ so $\Pr[\bar{i}_1\bar{i}_2] = 1 = (1-p_{1A})(1-q_{1A})$. Otherwise, if $X \in \{1_A, 1_B\}$, then at least one of q_X and p_X is 1, so all centers or leaves are opened, so $\Pr[\bar{i}_1\bar{i}_2] = 0$.

The remaining case is when $X = 2$. Notice that line 5 of Algorithm 17 is called when either $p_2 = 0$ or $p_2 = 1$. The only time $p_2 = 0$ is $\mathcal{A}_8$, in which $q_2 = 1$, so all the leaves are opened and $\Pr[\bar{i}_1\bar{i}_2] = 0$. If $p_2 = 1$, then i_1 is always opened and $\Pr[\bar{i}_1\bar{i}_2] = 0$. Otherwise $\mathcal{T}_2$ is divided into one group of large stars, and many groups $\mathcal{G}_s$ of small stars. Again, if i_1 and i_2 are in different groups, they are rounded independently. If they are both in a large star, then i_1 will always be opened and $\Pr[\bar{i}_1\bar{i}_2] = 0$. If they are both in the same small star, then the center-or-leaves property of our algorithm implies they will never both be closed, so $\Pr[\bar{i}_1\bar{i}_2] = 0$.

In the only remaining case, we have that ROUND2STARS is run (and $p_2 + q_2 = 1$), and i_1 and i_2 lie in separate stars within the same group $\mathcal{G}_s$. Let $\mathcal{E}$ be the event that "i_1 is among the c random facilities chosen to be opened in line 7 of

ROUND2STARS". We first show

$$\begin{aligned}
\Pr[X_{i_1} = 1 \wedge \bar{i_2}] &\le \sum_{x_{i_1} \in \mathcal{U}} x_{i_1} \Pr[X_{i_1} = x_{i_1} \wedge \bar{i_2}] \\
&= \sum_{x_{i_1} \in \mathcal{U}} x_{i_1} \sum_{x_{i_3} \in \mathcal{U}} \Pr[X_{i_1} = x_{i_1} \wedge \bar{i_2} \wedge X_{i_3} = x_{i_3}] \\
&= \sum_{x_{i_1} \in \mathcal{U}} x_{i_1} \sum_{x_{i_3} \in \mathcal{U}} \Pr[\bar{i_2} | X_{i_1} = x_{i_1} \wedge X_{i_3} = x_{i_3}] \Pr[X_{i_1} = x_{i_1} \wedge X_{i_3} = x_{i_3}] \\
&\le \sum_{x_{i_1} \in \mathcal{U}} x_{i_1} \sum_{x_{i_3} \in \mathcal{U}} (1 - x_{i_3}) \Pr[X_{i_1} = x_{i_1} \wedge X_{i_3} = x_{i_3}] \\
&= \mathbf{E}[X_{i_1}(1 - X_{i_3})].
\end{aligned}$$

If $|\mathcal{G}_s| \le c$, then all facilities in $\mathcal{G}_s$ will be opened and $\Pr[\bar{i_1}\bar{i_2}] = 0$. Otherwise, we can bound $\Pr[\bar{i_1}\bar{i_2}]$ as follows. Conditioned on $\bar{\mathcal{E}}$, i_1 is closed iff $X_{i_1} = 1$. Thus,

$$\begin{aligned}
\Pr[\bar{i_1}\bar{i_2}] &= \Pr[\mathcal{E}] \Pr[\bar{i_1}\bar{i_2} | \mathcal{E}] + (1 - \Pr[\mathcal{E}]) \Pr[\bar{i_1}\bar{i_2} | \bar{\mathcal{E}}] \\
&= \frac{c}{|\mathcal{G}_s|} \cdot 0 + \left(1 - \frac{c}{|\mathcal{G}_s|}\right) \Pr[\bar{i_1}\bar{i_2} | \bar{\mathcal{E}}] \\
&= \left(1 - \frac{c}{|\mathcal{G}_s|}\right) \Pr[X_{i_1} = 1 \wedge \bar{i_2} | \bar{\mathcal{E}}] \\
&= \left(1 - \frac{c}{|\mathcal{G}_s|}\right) \Pr[X_{i_1} = 1 \wedge \bar{i_2}] \\
&\le \left(1 - \frac{c}{|\mathcal{G}_s|}\right) \mathbf{E}[X_{i_1}(1 - X_{i_3})] \\
&\le \left(1 - \frac{c}{|\mathcal{G}_s|}\right) \left(1 + \frac{16}{3|\mathcal{G}_s|\beta^2}\right) (1 - p_2)(1 - q_2),
\end{aligned}$$

where we use the near-independence property of WEIGHTEDDEPROUND2 in the last inequality. (There are $t = 2$ variables of interest, $n = |\mathcal{G}_s|$ total variables, and

$\alpha = \min\{q_2, 1 - q_2\} = \beta$.)

We want to choose c such that $\left(1 - \frac{c}{|\mathcal{G}_s|}\right)\left(1 + \frac{16}{3|\mathcal{G}_s|\beta^2}\right) \leq 1 + \eta$, or equivalently,

$$c \geq \frac{16/(3\beta^2) - \eta|\mathcal{G}_s|}{1 + 16/(3|\mathcal{G}_s|\beta^2)}.$$

Therefore, our choice of $c = \lceil 16/(3\beta^2) \rceil$ implies that (3.8) holds true in all cases.

□

3.3.2.4 The nonlinear factor-revealing program

Now we will construct a nonlinear program which bounds the ratio between the total connection cost and the cost of the bi-point solution. We first introduce some necessary notation. Partition the clients into classes according to the types of stars in which $i_1(j)$ and $i_2(j)$ lie:

$$\mathcal{J}^{(X,Y)} := \{j \in \mathcal{J} \mid i_1(j) \in \mathcal{C}_X \wedge i_2(j) \in \mathcal{L}_Y\} \quad \forall X \in \{0, 1_A, 1_B, 2\}, Y \in \{1_A, 1_B, 2\}.$$

Furthermore, since we have multiple cost bounds available, we want to use the one which will be smallest for each client. Simply put, we want to try connecting the client to the closest facility first. To this end, we define subclasses for clients who are closer to either $i_1(j)$ or $i_2(j)$, respectively:

$$\mathcal{J}^{P(X,Y)} := \{j \in \mathcal{J}^{(X,Y)} \mid d_2(j) \leq d_1(j)\} \tag{3.9}$$

$$\mathcal{J}^{N(X,Y)} := \{j \in \mathcal{J}^{(X,Y)} \mid d_1(j) < d_2(j)\}. \tag{3.10}$$

For $(X, Y) = (1_B, 2)$, we define the subclasses slightly differently. This takes into account whether each client is closer to $i_0(j)$ or $i_3(j)$:

$$\mathcal{J}^{P(1_B,2)} := \{j \in \mathcal{J}^{(1_B,2)} \mid d_2 \leq d_1 \wedge d_1 + 2d_2 \leq d_1 + g(d_1 + d_2)\} \tag{3.11}$$

$$\mathcal{J}^{P'(1_B,2)} := \{j \in \mathcal{J}^{(1_B,2)} \mid d_2 \leq d_1 \wedge d_1 + g(d_1 + d_2) < d_1 + 2d_2\} \tag{3.12}$$

$$\mathcal{J}^{N(1_B,2)} := \{j \in \mathcal{J}^{(1_B,2)} \mid d_1 < d_2 \wedge d_1 + 2d_2 \leq d_1 + g(d_1 + d_2)\} \tag{3.13}$$

$$\mathcal{J}^{N'(1_B,2)} := \{j \in \mathcal{J}^{(1_B,2)} \mid d_1 < d_2 \wedge d_1 + g(d_1 + d_2) < d_1 + 2d_2\}. \tag{3.14}$$

Define the following set of classes, observing $\{\mathcal{J}^Z\}_{Z \in \mathcal{Z}}$ fully partitions the set of clients.

$$\mathcal{Z} = \{P'(1_B, 2), N'(1_B, 2)\} \cup \bigcup_{\substack{W \in \{P,N\} \\ X \in \{0,1_A,1_B,2\} \\ Y \in \{1_A,1_B,2\}}} \{W(X, Y)\}.$$

For each client class $Z \in \mathcal{Z}$, let $D_1^Z := \sum_{j \in \mathcal{J}^Z} d_1(j)$ and $D_2^Z := \sum_{j \in \mathcal{J}^Z} d_2(j)$, be the total cost contribution to D_1 or D_2, respectively, from clients in class J^Z. Then define the following:

$$C_{213}^Z := D_2^Z + (1 - q_Y)(D_1^Z - D_2^Z) + 2(1 - p_X)(1 - q_Y)D_2^Z$$

$$C_{123}^Z := D_1^Z + (1 - p_X)(D_2^Z - D_1^Z) + (1 - p_X)(1 - q_Y)(D_1^Z + D_2^Z)$$

$$C_{210}^Z := D_2^Z + (1 - q_Y)(D_1^Z - D_2^Z) + (1 - p_X)(1 - q_Y)g(D_1^Z + D_2^Z)$$

$$C_{120}^Z := D_1^Z + (1 - p_X)(D_2^Z - D_1^Z) + (1 - p_X)(1 - q_Y)g(D_1^Z - D_2^Z + g(D_1^Z + D_2^Z))$$

$$C_{145}^Z := D_1^Z + (1 - p_X)\left(2D_2^Z + \frac{1}{g}(D_1^Z + D_2^Z)\right) + (1 - p_X)(1 - q_2)\frac{1}{g}(D_1^Z + D_2^Z).$$

Finally, given an algorithm $\mathcal{A}_i = A(p_0, p_{1A}, q_{1A}, p_{1B}, q_{1B}, p_2, q_2)$, define

$$COST_1(\mathcal{A}_i) := C_{210}^{P'(1_B,2)} + C_{120}^{N'(1_B,2)} + \sum_{\substack{X\in\{0,1_A,1_B,2\}\\ Y\in\{1_A,1_B,2\}}} \left(C_{213}^{P(X,Y)} + C_{123}^{N(X,Y)}\right), \tag{3.15}$$

$$COST_2(\mathcal{A}_i) := C_{210}^{P'(1_B,2)} + C_{120}^{N'(1_B,2)} + \sum_{\substack{X\in\{0,1_A,1_B,2\}\\ Y\in\{1_B,2\}}} \left(C_{213}^{P(X,Y)} + C_{123}^{N(X,Y)}\right) + \sum_{X\in\{0,1_A,1_B,2\}} C_{145}^{(X,1_A)}. \tag{3.16}$$

Lemma 3.3.8. *For algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_7$ and $\mathcal{A}_9$, the total expected cost is bounded above by $(1+\eta)COST_1(\mathcal{A}_i)$. The expected cost of $\mathcal{A}_8$ is bounded above by $(1+\eta)COST_2(\mathcal{A}_8)$.*

Proof. Sum the bounds from Lemmas 3.3.4, 3.3.5, and 3.3.6 over each corresponding client class, and apply the bounds from Lemma 3.3.7. To apply those upper bounds, we need that the coefficients of $\Pr[\bar{i}_1]$, $\Pr[\bar{i}_1\bar{i}_2]$ (or similar terms) are nonnegative. This follows by definition of the class being summed over. (For example, for class $P(X,Y)$, we have $d_2 \le d_1$, so $d_1 - d_2 \ge 0$.) By linearity of expectation, we get the total expected cost of the algorithm. □

Our NLP: max $\quad X$ (3.17)

s.t $\quad X \leq COST_1(\mathcal{A}_i) \qquad \forall i \in \{1, 2, 3, 4, 5, 6, 7, 9\}$ (3.18)

$X \leq COST_2(\mathcal{A}_8)$ (3.19)

$D_2^Z \leq D_1^Z \qquad \forall Z = P(X, Y) \text{ or } Z = P'(1_B, 2)$ (3.20)

$D_2^Z \geq D_1^Z \qquad \forall Z = N(X, Y) \text{ or } Z = N'(1_B, 2)$ (3.21)

$(2 - g)D_2^{W(1_B,2)} \leq gD_1^{W(1_B,2)} \quad \forall W \in \{P, N\}$ (3.22)

$(2 - g)D_2^{W(1_B,2)} \geq gD_1^{W(1_B,2)} \quad \forall W \in \{P', N'\}$ (3.23)

$$\sum_{Z \in \mathcal{Z}} D_1^Z = \frac{1}{1 - b + br_D} \tag{3.24}$$

$$\sum_{Z \in \mathcal{Z}} D_2^Z = \frac{r_D}{1 - b + br_D} \tag{3.25}$$

$0.508 \leq b \leq 3/4$ (3.26)

$19/40 \leq r_D \leq 2/3$ (3.27)

$5/6 \leq s_0 \leq 1$

$g \geq 0$

$D_1^Z, D_2^Z \geq 0 \qquad \forall Z \in \mathcal{Z}$

Lemma 3.3.9. *Given a bi-point solution with cost $aD_1 + bD_2$ as input, with $s_0 \geq 5/6, b \in [0.508, 3/4], r_D \in [19/40, 2/3]$, and $r_1 > 1$, the best solution returned by*

$\mathcal{A}_1, \ldots, \mathcal{A}_9$ *has expected cost* $E[COST] \leq X^* \cdot (1+\eta)(aD_1 + bD_2)$*, where* X^* *is the solution to the above nonlinear program. Furthermore,* $X^* \in [1.3370,\ 1.3371]$.

Proof. Given a bi-point instance $a\mathcal{F}_1 + b\mathcal{F}_2$, first normalize all the distances by dividing by $aD_1 + bD_2$. This does not change the solution or the ratio of approximation obtained. Let X be the cost of the solution given by Algorithm 17. Because of the normalization, X is also the bi-point rounding factor. Constraints (3.18) and (3.19) must hold because we take the best cost of all algorithms. Lemma 3.3.8 shows that X may be a factor $(1+\eta)$ larger. Constraints (3.20), (3.21), (3.22), and (3.23) must hold by definition of each client class (see (3.9) through (3.14)). Constraints (3.24) and (3.25) enforce that the corresponding distance contributions from each client class sum to D_1 and D_2 (normalized).

We observe that for a fixed set of values of b, r_D, s_0, and g, the program becomes linear. We exploit this with computer-assisted methods (rigorous interval-arithmetic) and prove that $1.3370 \leq X^* \leq 1.3371$. □

3.3.3 Algorithms for edge cases

We have several border cases which we handle in a different, generally simpler, manner. We first prove these two facts:

Claim 3.3.2. $r_2 \leq 1/s_0$.

Proof. By definition of 2-stars, we have $|\mathcal{C}_2| \leq |\mathcal{L}_2|/2$ which implies $|\mathcal{C}_2| \leq (|\mathcal{C}_1| + |\mathcal{L}_2|) - (|\mathcal{C}_0| + |\mathcal{C}_1| + |\mathcal{C}_2|) + |\mathcal{C}_0| = \Delta_F + |\mathcal{C}_0|$. Thus, $r_2 \leq 1 + r_0 = 1/s_0$. □

Claim 3.3.3. $\frac{|\mathcal{L}_2|}{\Delta_F} \leq \frac{2}{s_0}$.

Proof. Since $|\mathcal{L}_2|/2 \le |\mathcal{L}_2| - |\mathcal{C}_2| = \Delta_F + |\mathcal{C}_0|$, we have $\frac{|\mathcal{L}_2|}{2\Delta_F} \le 1 + r_0 = 1/s_0$.

□

Lemma 3.3.10. *There is a* $(1+\eta)\cdot 1.3371$*-approximation algorithm for rounding the bi-point solution and opens at most* $k + O(\log(1/\eta))$ *facilities when either* $b \le 0.508,\ b \ge 3/4,\ r_D \le 19/40,$ *or* $r_D \ge 2/3$.

Proof. Basically, we just return the better solutions between $\mathcal{F}_1$ and the one produced by $\mathcal{A}' = \mathcal{A}(a, a, b, a, b, a, b)$. As mentioned before, $\mathcal{A}'$ and Li-Svensson's rounding algorithm are essentially the same, and output a solution with the same upper-bound on the expected cost:

$$(1+\eta)(ad_1 + b(1+2a)d_2),$$

The main difference is that $\mathcal{A}'$ only opens at most $O(\log(1/\eta))$ (instead of $O(1/\eta)$) extra facilities.

As in [35], we need to be careful when a or b is close to 0 because the number of extra facilities opened by $\mathcal{A}'$ is roughly $\frac{16}{3\beta^2}\log_{1+\beta}(1/(a\cdot\eta))$, where $\beta = \min\{a, b\}$. We consider two corner cases:

- If $0 \le b \le 1/4$, we can just return $\mathcal{F}_1$ as our solution. Note that $|\mathcal{F}_1| \le k$ and the approximation ratio is $\frac{d_1}{ad_1+bd_2} \le \frac{1}{a} = \frac{1}{1-b} \le 4/3$.

- If $b \ge 5/6$, we can use the knapsack algorithm described in [35], which only opens at most $k+2$ facilities, to get a 4/3-approximation algorithm. Note that the approximation ratio of this algorithm is bounded by $1 + 2a \le 4/3$ as

$a \le 1/6$.

- We claim that the above algorithm gives an approximation ratio of 1.337 for all remaining cases. Note that the cost of this algorithm is at most $(1+\eta)\min\{d_1, ad_1 + b(1+2a)d_2\}$. Thus, it suffices to bound the ratio

$$\begin{aligned} f &:= \frac{\min\{d_1, ad_1 + b(1+2a)d_2\}}{ad_1 + bd_2} \\ &= \min\left\{\frac{1}{1-b+br_D}, \frac{1-b+b(1+2(1-b))r_D}{1-b+br_D}\right\}. \end{aligned}$$

Note that the right-hand-side is a function of b and r_D. When $b \in [1/4, 0.508]$, $b \in [3/4, 5/6]$, $r_D \le 19/40$, or $r_D \ge 2/3$, that function is at most 1.337 by elementary calculus.

Observe that

$$\frac{\partial}{\partial r_D}\left(\frac{1}{1-b+br_D}\right) = -\frac{b}{(1-b+br_D)^2} \le 0,$$

and

$$\frac{\partial}{\partial r_D}\left(\frac{1-2b^2 r_D + b(-1+3r_D)}{1-b+br_D}\right) = \frac{2(-1+b)^2 b}{(1+b(-1+r_D))^2} \ge 0.$$

It means that, for a fixed value of b, the former is a decreasing function of r_D and the latter is an increasing function of r_D. Therefore,

  - Case $b \in [1/4, 0.508]$ or $b \in [3/4, 5/6]$: The maximum of f will be achieved at some point such that

$$\frac{1}{1-b+br_D} = \frac{1-2b^2r_D+b(-1+3r_D)}{1-b+br_D},$$

or equivalently,

$$r_D = \frac{1}{3-2b}.$$

Then, in this case,

$$f \leq \max_{b\in[1/4,0.508]\cup[3/4,5/6], r_D=\frac{1}{3-2b}} \frac{1}{1-b+br_D} = 1.33681.$$

– Case $b \in [0.508, 3/4]$ and $r_D \leq 19/40$: Note that $r_D \leq \frac{1}{3-2b}$ which implies that $\frac{1}{1-b+br_D} \geq \frac{1-2b^2r_D+b(-1+3r_D)}{1-b+br_D}$. Since the RHS is increasing in r_D, we have

$$f \leq \max_{b\in[0.508,3/4], r_D=19/40} \frac{1-2b^2r_D+b(-1+3r_D)}{1-b+br_D} = 1.33294.$$

– Case $b \in [0.508, 3/4]$ and $r_D \geq 2/3$: Note that $r_D \geq \frac{1}{3-2b}$ which implies that $\frac{1}{1-b+br_D} \leq \frac{1-2b^2r_D+b(-1+3r_D)}{1-b+br_D}$. Since the LHS is decreasing in r_D, we have

$$f \leq \max_{b\in[0.508,3/4], r_D=2/3} \frac{1}{1-b+br_D} = \frac{4}{3} \leq 1.33334.$$

□

Lemma 3.3.11. *There is a* $(1+\eta)\cdot 1.3371$*-approximation algorithm for rounding the* *bi-point solution which opens at most $k + O(\log(1/\eta))$ facilities when $s_0 \leq 5/6, b \in [0.508, 3/4]$, and $r_D \in [19/40, 2/3]$.*

Proof. We show that the better solution returned from a set of 3 algorithms will be within a factor 1.3371 of the optimal solution. The purpose of this case is to bound s_0 away from 0, so that p_2 and q_2 in our main case are bounded away from zero. The first algorithm is a knapsack algorithm in which we open all facilities in $\mathcal{L}_1$ and $\mathcal{C}_2$. After that we almost greedily choose some of the 2-stars, close their centers, and open all their leaves. This algorithm does very well if s_0 is small. In the second algorithm, we open $\mathcal{F}_1$ and some additional facilities in $\mathcal{L}_2$ which maximize the saving. In particular, we use the following algorithms:

- Algorithm 1: Open all facilities in $\mathcal{L}_1$ and $\mathcal{C}_2$. For each client j, if $i_2(j) \in \mathcal{L}_1$, connect j to $i_2(j)$. Otherwise $i_2(j)$ is a leaf of a 2-star, let i_3 be the center of this star and connect j to i_3. Thus, the total connection cost of the current solution is upper-bounded by

  $$\sum_{j:i_2(j)\in\mathcal{L}_1} d_2(j) + \sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j) + 2d_2(j)) = D_2 + \sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j) + d_2(j)).$$

  Now, for each $i \in \mathcal{C}_2$, if we close facility i and open all of its leaves, the total cost will be reduced by $\sum_{j\in\delta(\mathcal{S}_i)}(d_1(j) + d_2(j))$, where $\delta(\mathcal{S}_i)$ is the set of clients j having $i_2(j) \in \mathcal{S}_i$, and we also open additional $|\mathcal{S}_i| - 1$ facilities. This motivates us to solve the following knapsack LP, just as in [35]:

$$\text{maximize} \sum_{i\in\mathcal{C}_2} x_i \left(\sum_{j\in\delta(\mathcal{S}_i)} (d_1(j)+d_2(j)) \right)$$

$$\text{subject to} \sum_{i\in\mathcal{C}_2} x_i(|\mathcal{S}_i|-1) \le k - |\mathcal{L}_1| - |\mathcal{C}_2|$$

$$0 \le x_i \le 1 \ \forall i \in \mathcal{C}_2$$

Note that a basic solution of the above LP only has at most 1 fractional value. Thus, we can easily obtain it by a greedy method. Let us call this fractional value x_{i^*}. Now, for all $i \in \mathcal{C}_2$, if $x_i = 0$, we keep i opened. If $x_i = 1$, we close i and open all of its leaves. We also open i^* and a subset of size $\lceil x_{i^*}|\mathcal{S}_{i^*}|\rceil$ of $\mathcal{S}_{i^*}$ uniformly at random. It is easy to see that the expected saved cost is at least the optimal value of the LP by doing so.

- Algorithm 2: Open all facilities in $\mathcal{F}_1$. Define the saving of a facility $i \in \mathcal{L}_2$ be $\sum_{j\in\delta(\mathcal{S}_i)}(d_1(j)-d_2(j))_+$. Sort all the facilities in $\mathcal{L}_2$ non-increasing by its saving. Open the first $\lceil \frac{bs_0}{2}|\mathcal{L}_2|\rceil$ facilities in this order.

Analysis:

- The two algorithms only open $k+2$ facilities. In the first algorithm, we claim that at most $k+2$ facilities will be opened. The first constraint of the LP guarantee that a fractional solution x will open at most k facilities. The two extra facilities come from the fact that we open i^* and take the ceiling of

$x_{i^*}|\mathcal{S}_{i^*}|$. In the second algorithm, we open at most

$$\begin{aligned}|\mathcal{F}_1| + \frac{bs_0}{2}|\mathcal{L}_2| + 1 &\le |\mathcal{F}_1| + \frac{b|\mathcal{L}_2|}{2} \times \frac{2\Delta_F}{|\mathcal{L}_2|} + 1 \\ &= |\mathcal{F}_1| + b\Delta_F + 1 = k + 1,\end{aligned}$$

where the first inequality is due to $s_0 \le \frac{2\Delta_F}{|\mathcal{L}_2|}$, by Claim 3.3.3.

- Now, we bound the cost of the first algorithm. Let q be the maximum value such that the solution $x_i = q$ for all $i \in C_2$ is feasible to the knapsack LP. We solve for q by requiring

$$\begin{aligned}&\sum_{i\in\mathcal{C}_2} q(|\mathcal{S}_i| - 1) \le k - |\mathcal{L}_1| - |\mathcal{C}_2| \\ &\iff q(|\mathcal{L}_2| - |\mathcal{C}_2|) \le k - |\mathcal{L}_1| - |\mathcal{C}_2| \\ &\iff q \le \frac{k - |\mathcal{L}_1| - |\mathcal{C}_2|}{|\mathcal{L}_2| - |\mathcal{C}_2|} = \frac{|\mathcal{F}_1| + b\Delta_F - |\mathcal{L}_1| - |\mathcal{C}_2|}{|\mathcal{L}_2| - |\mathcal{C}_2|} = \frac{b\Delta_F + |\mathcal{C}_0|}{|\mathcal{L}_2| - |\mathcal{C}_2|}.\end{aligned}$$

Thus, we can set $q := \frac{b\Delta_F + |\mathcal{C}_0|}{|\mathcal{L}_2| - |\mathcal{C}_2|}$. Note that $|\mathcal{L}_2| - |\mathcal{C}_2| = |\mathcal{C}_0| + \Delta_F$, we have

$$q = \frac{b\Delta_F + |\mathcal{C}_0|}{|\mathcal{C}_0| + \Delta_F} = \frac{b + r_0}{1 + r_0} = 1 - as_0.$$

Since $x = q$ is a feasible solution, the saved cost is at least

$$\sum_{i\in\mathcal{C}_2} q\left(\sum_{j\in\delta(\mathcal{S}_i)} (d_1(j) + d_2(j))\right) = (1 - as_0) \sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j) + d_2(j)).$$

Therefore, we can upper-bound the cost by

$$\begin{aligned} & D_2 + \sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j)+d_2(j)) - (1-as_0) \sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j)+d_2(j)) \\ & = D_2 + as_0 \sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j)+d_2(j)). \end{aligned}$$

- Recall that the sets $\delta(\mathcal{S}_i)$ are pairwise disjoint. By a simple average argument, the cost of the second algorithm is upper-bounded by

$$D_1 - \frac{bs_0}{2} \sum_{i\in\mathcal{L}_2} \sum_{j\in\delta(\mathcal{S}_i)} (d_1(j)-d_2(j))_+ \le D_1 - \frac{bs_0}{2} \left(\sum_{j:i_2(j)\in\mathcal{L}_2} (d_1(j)-d_2(j)) \right)_+ .$$

We run these two algorithms along with $\mathcal{A}' = \mathcal{A}(a,a,b,a,b,a,b)$, and use the best solution of the three. (Again, note that $\beta = \min\{a,b\} \ge 1/4$ and $a \ge 1/4$ in this case.) We can easily formulate an NLP to derive the approximation ratio as discussed in subsection 3.3.2.3. Using an interval search as before over the interval $0 \le s_0 \le 5/6, b \in [0.508, 3/4], r_D \in [19/40, 2/3]$, we get an upper-bound of 1.3371 on the factor-revealing NLP. Note that the value of g is irrelevant to any of these algorithms, so our intervals are over only b, r_D and s_0. This interval search runs in seconds and examines about 6400 intervals.

□

Lemma 3.3.12. *There is a* $(1+\eta)\cdot 1.3371$*-approximation algorithm for rounding the bi-point solution which opens at most* $k + O(\log(1/\eta))$ *facilities when* $s_0 \ge 5/6, b \in [0.508, 3/4], r_D \in [19/40, 2/3]$, *and* $r_1 \le 1$.

Proof. We will run the set of 10 algorithms shown in Table 3.2. Obviously, when ROUND2STARS is called (only algorithms $\mathcal{A}'_1, A'_2, \mathcal{A}'_7, \mathcal{A}'_8$), we have $\beta = \min\{q_2, 1-q_2\} \geq 5/24$ and $p_2 \geq 5/24$, which are achieved at $p_2 = as_0$, $a = 1/4$, $s_0 = 5/6$. Thus, it is easy to check that $\mathcal{A}'_1$, A'_2, and $A'_4, \ldots, \mathcal{A}'_8$ only open $k + O(\log(1/\eta))$ facilities. For $\mathcal{A}'_9$ and $\mathcal{A}'_{10}$, using the same argument as in Lemma 3.3.3 and Lemma 3.3.2, we open at most

$$|\mathcal{F}_1| + b|\mathcal{C}_1| + O(\log(1/\eta)) \leq |\mathcal{F}_1| + (b/r_1)|\mathcal{C}_1| + O(\log(1/\eta))$$
$$= |\mathcal{F}_1| + b\Delta_F + O(\log(1/\eta)) = k + O(\log(1/\eta)),$$

where the first inequality is due to the fact that $r_1 \leq 1$ in the interval of interest.

Recall that $s_0 = \frac{1}{1+r_0} = \frac{1}{1+|\mathcal{C}_0|/\Delta_F} = \frac{|\mathcal{L}_2|-|\mathcal{C}_2|-|\mathcal{C}_0|}{|\mathcal{L}_2|-|\mathcal{C}_2|} = 1 - \frac{|\mathcal{C}_0|}{|\mathcal{L}_2|-|\mathcal{C}_2|}$. For $\mathcal{A}'_3$, we open at most

$$\begin{aligned}
|\mathcal{C}_1| + |\mathcal{C}_2| + \frac{1-as_0}{2}|\mathcal{L}_2| + O(\log(1/\eta)) &= |\mathcal{C}_1| + |\mathcal{C}_2| + \frac{1-a+a\frac{|\mathcal{C}_0|}{|\mathcal{L}_2|-|\mathcal{C}_2|}}{2}|\mathcal{L}_2| + O(\log(1/\eta)) \\
&= |\mathcal{C}_1| + |\mathcal{C}_2| + \frac{b}{2}|\mathcal{L}_2| + a|\mathcal{C}_0|\frac{|\mathcal{L}_2|}{2(|\mathcal{L}_2|-|\mathcal{C}_2|)} + O(\log(1/\eta)) \\
&\leq |\mathcal{C}_1| + |\mathcal{C}_2| + \frac{b}{2}|\mathcal{L}_2| + a|\mathcal{C}_0| + O(\log(1/\eta)) \\
&= |\mathcal{C}_1| + a|\mathcal{C}_2| + b|\mathcal{C}_2| + b|\mathcal{L}_2| - \frac{b}{2}|\mathcal{L}_2| + a|\mathcal{C}_0| \\
&\qquad + O(\log(1/\eta)) \\
&\leq |\mathcal{C}_1| + a|\mathcal{C}_2| + b|\mathcal{L}_2| + a|\mathcal{C}_0| + O(\log(1/\eta)) \\
&= k + O(\log(1/\eta)).
\end{aligned}$$

The approximation ratio will be bounded by an NLP as in our main case. It is simpler, as we need not consider the distinction between $\mathcal{T}_{1A}$ and T_{1B}, or the value of g. We do an interval search and get an upper-bound of 1.337 when $b \in [0.508, 3/4], r_D \in [19/40, 2/3]$, and $s_0 \in [5/6, 1]$.

Algorithms	p_0	p_{1A}	q_{1A}	p_{1B}	q_{1B}	p_2	q_2
$\mathcal{A}'_1$	0	0	1	0	1	as_0	$1 - as_0$
$\mathcal{A}'_2$	0	1	0	1	0	as_0	$1 - as_0$
$\mathcal{A}'_3$	0	0	1	0	1	1	$\frac{1-as_0}{2}$
$\mathcal{A}'_4$	0	1	0	1	0	1	$\frac{1-as_0}{2}$
$\mathcal{A}'_5$	1	0	1	0	1	1	$\frac{bs_0}{2}$
$\mathcal{A}'_6$	1	1	0	1	0	1	$\frac{bs_0}{2}$
$\mathcal{A}'_7$	1	0	1	0	1	$1 - bs_0$	bs_0
$\mathcal{A}'_8$	1	1	0	1	0	$1 - bs_0$	bs_0
$\mathcal{A}'_9$	1	1	b	1	b	1	0
$\mathcal{A}'_{10}$	1	b	1	b	1	1	0

Table 3.2: Calls of $\mathcal{A}$ when $r_1 \leq 1$.

$\square$

The main result in this section is summarized in the following theorem.

Theorem 3.3.1. *There is a* $(1 + \eta) \cdot 1.3371$*-approximation algorithm for rounding the bi-point solution which opens at most* $k + O(\log(1/\eta))$ *facilties.*

3.4 Dichotomy result

In the last subsections, we introduced a $(2.675 + \epsilon)$-approximation algorithm for the k-median problem which runs in $n^{O((1/\epsilon)\log(1/\epsilon))}$ time. Now we show that by using a simple scaling technique and careful analysis, we can either improve the runtime by getting rid of the $\log(1/\epsilon)$ factor in the power of n, or we can improve the approximation ratio. Our result is summarized in the following theorem. Recall from the last subsection that, when ROUND2STARS(p_2, q_2) is called, $\beta := \min\{q_2, 1-q_2\}$ is strictly bounded away from zero.

Theorem 3.4.1. *For any parameter $\epsilon > 0$ small enough, there exist algorithms A_ϵ and B_ϵ such that, for any instance $\mathcal{I}$ of the k-median problem, either A_ϵ is fast or B_ϵ is more accurate:*

- *A_ϵ is a randomized $(2.675 + \epsilon)$-approximation algorithm which produces a solution to $\mathcal{I}$ with constant probability and runs in $n^{O(1/\epsilon)}$ time, or*
- *B_ϵ is a $(2 + \epsilon)$-approximation algorithm for $\mathcal{I}$ which runs in $n^{O(\text{poly}(1/\epsilon))}$ time.*

We say that a star $\mathcal{S}_i$ with $i \in \mathcal{C}_2$ is small if $2 \le |\mathcal{S}_i| \le \frac{c_0}{\eta}$ for some constant $c_0 > 0$. Otherwise, $|\mathcal{S}_i| > \frac{c_0}{\eta}$ and we call it a large star. Again, let $\mathcal{C}_2'$ and $\mathcal{L}_2'$ denote sets of centers and leaves of large stars. Also let $\mathcal{C}_2''$ and $\mathcal{L}_2''$ be sets of centers and leaves of small stars.

First, observe that for large stars, we can reuse the following trick: move a little mass from the leaves to open the center. In other words, we will open C_2' and a subset of size $\lceil q_2(|\mathcal{L}_2'| - |\mathcal{C}_2'|)\rceil$ of $\mathcal{L}_2'$. For $i_2 \in \mathcal{L}_2'$, it is not hard to show that $\Pr[i_2] \ge q - p\eta$ (i.e. the loss is negligible). We open 1 extra facility in this class. Recall that $\mathcal{A}$ opens at most 4 extra facilities in $\mathcal{T}_0 \cup \mathcal{T}_1 \cup \mathcal{C}_2' \cup \mathcal{L}_2'$. The question is can we also reduce the number of extra opened facilities which are part of small stars (previously, this number was $O(\log(1/\eta))$)? We consider the following cases.

- Case 1: $|\mathcal{C}_2''| > f(1/\eta)$ for some function $f = O(\text{poly}(1/\eta))$ to be determined.

  In this case, we have a lot of small stars. We scale down the probability of opening the leaves by $(1-\eta)$ and open/close the centers/leaves independently. That is, for each center $i \in \mathcal{C}_2''$, we randomly open $\mathcal{S}_i$ and close center i with probability $(1-\eta)q_2$. (With the remaining probability, we close $\mathcal{S}_i$ and open center i.) We show that, with constant probability, the algorithm returns a feasible solution whose cost is only blown up by a small factor of $(1+\eta)$.

- Case 2: $|\mathcal{L}_2'| + |\mathcal{L}_2''| \leq g(1/\eta)$ for some function $g = O(\text{poly}(1/\eta))$ to be determined. In this case, the number of leaves should be small enough so that we can simply open all the leaves in $\mathcal{L}_2$. The number of extra opened facilities is $O(g(1/\eta))$. However, we achieve a pseudo solution with no loss in connection cost compared to the bipoint solution.

- Case 3: Neither Case 1 nor Case 2 holds (i.e. $|\mathcal{C}_2''| \leq f(1/\eta)$ and $|\mathcal{L}_2'| + |\mathcal{L}_2''| \geq g(1/\eta)$). Note that, by definition of small stars,

$$|\mathcal{L}_2''| \leq \frac{c_0}{\eta}|\mathcal{C}_2''| \leq \frac{c_0 f(1/\eta)}{\eta}.$$

  This implies that

$$|\mathcal{L}_2'| \geq g(1/\eta) - |\mathcal{L}_2''| \geq g(1/\eta) - \frac{c_0 f(1/\eta)}{\eta}.$$

  Intuitively, the number of centers and leaves of small stars are upper-bounded by some constant. On the other hand, we have a lower-bound on the number of leaves of large stars. If we have enough leaves in $\mathcal{L}_2'$, we can scale down the probability to open each facility in $\mathcal{L}_2'$ so that all centers in $\mathcal{C}_2''$ can be opened without violation.

Details of these cases will be showed in the following subsections.

3.4.1 Case 1

For each $i \in \mathcal{C}_2''$, let X_i be an indicator of the event that we open $\mathcal{S}_i$ and close i. (If $X_i = 0$, we close S_i and open i.) The idea is to set each $X_i = 1$ independently with probability $(1-\eta)q_2$.

Lemma 3.4.1. *With probability at least* $1 - \exp\left(-\frac{\eta^3(1-\eta)\beta f(1/\eta)}{3c_0}\right)$, *the algorithm opens at most* $p_2|\mathcal{C}_2''| + q_2|\mathcal{L}_2''|$ *facilities which are part of small stars.*

Proof. Recall that small stars have at most c_0/η leaves. The number of opened facilities which are part of small stars is

$$X = \sum_{i \in \mathcal{C}_2''} (X_i|\mathcal{S}_i| + (1 - X_i)) = |\mathcal{C}_2''| + \sum_{i \in \mathcal{C}_2''} X_i(|\mathcal{S}_i| - 1) = |\mathcal{C}_2''| + \frac{c_0}{\eta} \sum_{i \in \mathcal{C}_2''} Y_i,$$

where $Y_i = \frac{X_i(|\mathcal{S}_i|-1)}{c_0/\eta}$. Note that Y_i's take random values in $[0,1]$. The expected value of $Y = \sum_{i \in \mathcal{C}_2''} Y_i$ is

$$\begin{aligned}
\mu &:= \mathbf{E}\left[\sum_{i \in \mathcal{C}_2''} Y_i\right] \\
&= \sum_{i \in \mathcal{C}_2''} \frac{\mathbf{E}[X_i](|\mathcal{S}_i| - 1)}{c_0/\eta} \\
&= \frac{1-\eta}{c_0/\eta} \sum_{i \in \mathcal{C}_2''} q_2(|\mathcal{S}_i| - 1) \\
&= \frac{\eta(1-\eta)}{c_0} q_2(|\mathcal{L}_2''| - |\mathcal{C}_2''|) \\
&\geq \frac{\eta(1-\eta)}{c_0} q_2|\mathcal{C}_2''| \geq \frac{\eta(1-\eta)}{c_0}\beta f(1/\eta),
\end{aligned}$$

since each small star has at least 2 leaves. Using Chernoff's bound, we have

$$\begin{aligned}
\Pr\left[X > p_2|\mathcal{C}_2''| + q_2|\mathcal{L}_2''|\right] &= \Pr\left[|\mathcal{C}_2''| + \frac{c_0}{\eta} Y > |\mathcal{C}_2''| + q_2(|\mathcal{L}_2''| - |\mathcal{C}_2''|)\right] \\
&= \Pr\left[Y > \frac{\eta}{c_0} q_2(|\mathcal{L}_2''| - |\mathcal{C}_2''|)\right] \\
&= \Pr\left[Y > \frac{\mu}{1-\eta}\right] \\
&\leq \Pr\left[Y > (1+\eta)\mu\right] \\
&\leq \exp\left(-\frac{\eta^2}{3}\mu\right) \\
&\leq \exp\left(-\frac{\eta^3(1-\eta)\beta f(1/\eta)}{3c_0}\right).
\end{aligned}$$

□

To bound the expected connection cost, we need the following lemma.

Lemma 3.4.2. *Let i_1 and i_2 be any facilities in $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Let $X, Y \in$*

$\{0, 1_A, 1_B, 2\}$ *be the classes such that* $i_1 \in \mathcal{C}_X$ *and* $i_2 \in \mathcal{L}_Y$. *Then the following are true:*

$$\Pr[\bar{i_1}] \le 1 - p_X, \tag{3.28}$$

$$\Pr[\bar{i_2}] \le \left(1 + \frac{1-\beta}{\beta}\eta\right)(1 - q_Y), \tag{3.29}$$

$$\Pr[\bar{i_1}\bar{i_2}] \le \left(1 + \frac{1-\beta}{\beta}\eta\right)(1 - p_X)(1 - q_Y). \tag{3.30}$$

Proof. The proof is quite similar to that of lemma 3.3.7.

- Proof of (3.28): We only need to check the case $i_1 \in \mathcal{C}_2''$. It is clear that

$$\Pr[\bar{i_1}] = (1-\eta)q_2 = 1 - p_2 - \eta q_2 \le 1 - p_2.$$

- Proof of (3.29): We only need to check the case $i_2 \in \mathcal{L}_2''$. It is clear that

$$\Pr[\bar{i_2}] = 1 - (1-\eta)q_2 \le \left(1 + \frac{1-\beta}{\beta}\eta\right)(1 - q_2),$$

  since $q_2 \le 1 - \beta$.

- Proof of (3.30): i_1 and i_2 are always opened independently

$$\Pr[\bar{i_1}\bar{i_2}] = \Pr[\bar{i_1}]\Pr[\bar{i_2}] \le \left(1 + \frac{1-\beta}{\beta}\eta\right)(1 - p_X)(1 - q_Y).$$

□

Corollary 3.4.1. *The expected connection cost of the solution returned by the algorithm is at most* $1.337 \cdot \left(1 + \frac{1-\beta}{\beta}\eta\right)$ *times the cost of the bipoint solution.*

Theorem 3.4.2. *There exists a choice of* $f = O(\text{poly}(1/\eta))$ *so that, when* $|\mathcal{C}_2''| > f(1/\eta)$*, the algorithm returns a solution opening at most 4 additional facilities and having connection cost at most* $1.3371 \cdot \left(1 + \frac{1-\beta}{\beta}\eta\right)(1+\eta)$ *times the cost of the bipoint solution with constant positive probability which is a function of* η*.*

Proof. Let $f(1/\eta) = \frac{3c_0}{\eta^3(1-\eta)\beta} \ln \eta^{-2}$. Also let $\mathcal{E}_1$ be the event "the connection cost is at most $1.3371 \cdot \left(1 + \frac{1-\beta}{\beta}\eta\right)(1+\eta)$ times the cost of the bipoint solution" and $\mathcal{E}_2$ be the event "the algorithm opens at most 4 extra facilities". By Markov bound,

$$\Pr[\mathcal{E}_1] \geq 1 - \frac{1}{1+\eta}.$$

Note that at most 4 additional facilities in $\mathcal{C}_2' \cup \mathcal{L}_2' \cup \mathcal{T}_0 \cup \mathcal{T}_{1A} \cup \mathcal{T}_{1B}$ could be opened. By the choice of f and lemma 3.4.1,

$$\Pr[\mathcal{E}_2] \geq 1 - \eta^2.$$

Thus,

$$\begin{aligned}
\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2] &= \Pr[\mathcal{E}_2] - \Pr[\bar{\mathcal{E}}_1 \wedge \mathcal{E}_2] \\
&\geq \Pr[\mathcal{E}_2] - \Pr[\bar{\mathcal{E}}_1] \\
&\geq (1-\eta^2) - \frac{1}{1+\eta} = 1 - \frac{\eta^3 + \eta^2 + 1}{\eta + 1},
\end{aligned}$$

which is strictly greater than zero when η is small enough. $\square$

3.4.2 Case 2

Assume $|\mathcal{L}'_2| + |\mathcal{L}''_2| \leq g(1/\eta)$ for some $g = O(\text{poly}(1/\eta))$ to be determined, we simply open all the facilities in $\mathcal{F}_2$. Indeed the number of extra opened facilities is $O(\text{poly}(1/\eta))$; however, the solution has cost equal to $\mathcal{D}_2 < a\mathcal{D}_1 + b\mathcal{D}_2$.

3.4.3 Case 3

If neither Case 1 nor Case 2 holds, we have $|\mathcal{C}''_2| \leq f(1/\eta)$ and $|\mathcal{L}'_2| + |\mathcal{L}''_2| \geq g(1/\eta)$. Since $|\mathcal{L}''_2| \leq \frac{c_0}{\eta}|\mathcal{C}''_2|$, we can bound the number of leaves of large stars

$$|\mathcal{L}'_2| \geq g(1/\eta) - |\mathcal{L}''_2| \geq g(1/\eta) - \frac{c_0 f(1/\eta)}{\eta}.$$

The "budget" to open facilities in the class of large stars is

$$p_2|\mathcal{L}'_2| + q_2|\mathcal{C}'_2| = |\mathcal{C}'_2| + q_2(|\mathcal{L}'_2| - |\mathcal{C}'_2|).$$

Suppose that we want to open each leaf in $\mathcal{L}'_2$ with probability $q_2(1 - c_1\eta)$ for some constant $c_1 \geq 1$ and open all the centers in $\mathcal{C}'_2$. Then the remaining budget which can be used to open other facilities in $\mathcal{C}''_2$ is

$$\begin{aligned} R &= |\mathcal{C}'_2| + q_2(|\mathcal{L}'_2| - |\mathcal{C}'_2|) - (|\mathcal{C}'_2| + q_2(1 - c_1\eta)|\mathcal{L}'_2|) \\ &= c_1\eta q_2|\mathcal{L}'_2| - q_2|\mathcal{C}'_2|. \end{aligned}$$

To open all centers in $\mathcal{C}_2''$, we need to open at most $q_2|\mathcal{C}_2''| \leq q_2 f(1/\eta)$ additional facilities in $\mathcal{C}_2''$, apart from the usual $p_2|\mathcal{C}_2''|$ ones. Thus, it suffices to require that $R \geq q_2 f(1/\eta)$.

- If $|\mathcal{C}_2'| \geq \frac{f(1/\eta)}{-1+c_0c_1}$, recall that $|\mathcal{L}_2'| \geq \frac{c_0}{\eta}|\mathcal{C}_2'|$, we have

$$\begin{aligned} R &\geq c_0c_1q_2|\mathcal{C}_2'| - q_2|\mathcal{C}_2'| \\ &\geq (c_0c_1 - 1)q_2\frac{f(1/\eta)}{-1+c_0c_1} \\ &= q_2 f(1/\eta). \end{aligned}$$

- Else $|\mathcal{C}_2'| < \frac{f(1/\eta)}{-1+c_0c_1}$, recall that $|\mathcal{L}_2'| \geq g(1/\eta) - \frac{c_0 f(1/\eta)}{\eta}$, we can lower-bound R as follows.

$$\begin{aligned} R &\geq c_1\eta q_2|\mathcal{L}_2'| - q_2\frac{f(1/\eta)}{-1+c_0c_1} \\ &\geq c_1\eta q_2\left(g(1/\eta) - \frac{c_0 f(1/\eta)}{\eta}\right) - q_2\frac{f(1/\eta)}{-1+c_0c_1}. \end{aligned}$$

A simple calculation shows that we can choose

$$g(1/\eta) = \frac{c_0^2c_1}{\eta(c_0c_1-1)}f(1/\eta),$$

then

$$\begin{aligned}
R &\geq c_1 \eta q_2 \left(\frac{c_0^2 c_1}{\eta(c_0 c_1 - 1)} f(1/\eta) - \frac{c_0 f(1/\eta)}{\eta} \right) - q_2 \frac{f(1/\eta)}{-1 + c_0 c_1} \\
&= \frac{c_0^2 c_1^2}{c_0 c_1 - 1} q_2 f(1/\eta) - c_0 c_1 q_2 f(1/\eta) - \frac{1}{c_0 c_1 - 1} q_2 f(1/\eta) \\
&= \left(\frac{c_0^2 c_1^2}{c_0 c_1 - 1} - c_0 c_1 - \frac{1}{c_0 c_1 - 1} \right) q_2 f(1/\eta) \\
&= \left(\frac{c_0^2 c_1^2 - c_0 c_1 (c_0 c_1 - 1) - 1}{c_0 c_1 - 1} \right) q_2 f(1/\eta) \\
&= q_2 f(1/\eta).
\end{aligned}$$

Theorem 3.4.3. *For any polynomial function $f(1/\eta)$, let*

$$g(1/\eta) = \frac{2}{\eta} f(1/\eta).$$

If $|\mathcal{C}_2''| \leq f(1/\eta)$ and $|\mathcal{L}_2'| + |\mathcal{L}_2''| \geq g(1/\eta)$, then there is an algorithm which returns a solution opening at most 4 additional facilities and having expected connection cost at most $1.337 \cdot (1 + \frac{1-\beta}{\beta}\eta)(1+\eta)$ times the cost of the bipoint solution.

Proof. We simply set $c_0 = 2$ and $c_1 = 1$. As discussed above, there are no extra opened facilities in $\mathcal{C}_2'' \cup \mathcal{L}_2''$. Lemma 3.4.2 also holds in this case and can be used to bound the expected connection cost. □

This implies an approximation algorithm which runs in $n^{O(1/\epsilon)}$ time for k-median.

Chapter 4: The Knapsack Median Problem

4.1 Problem definition

In this chapter, we study a natural generalization of the k-median problem, known as the Knapsack Median (KM) problem, in which we have a non-negative weight w_i for each facility $i \in \mathcal{F}$, and instead of opening k facilities, we require the sum of weights of the open facilities to be at most a given budget $B \in \mathbb{R}_+$. In other words, we would like find a set $\mathcal{S} \subseteq \mathcal{F}$ such that (a) $\sum_{j \in \mathcal{C}} \min\{d(i,j) : i \in \mathcal{S}\}$ is minimized and (b) $\sum_{i \in \mathcal{S}} w_i \leq B$. We shall refer to any facility $\mathcal{S}$ as an *open* facility.

4.2 Prior work and Our contributions

While the KM problem is not known to be harder than k-MEDIAN, it has thus far proved more difficult to approximate. The k-median problem was first approximated to within a constant factor $6\frac{2}{3}$ in 1999 [33], with a series of improvements leading to the current best-known factor of 2.674 in this work.

The KM problem was first studied in 2011 by Krishnaswamy et. al. [51], who gave a bicriteria $16 + \epsilon$ approximation which slightly violated the budget by a factor of $(1 + \epsilon)$. Then Kumar gave the first true constant factor approximation for the

KM problem with a very large approximation ratio (about 2700) [52], subsequently reduced to 34 by Charikar & Li [53] and then to 32 by Swamy [12].

Our main contribution in this section is a 17.46-approximation algorithm for the KM problem. This algorithm has a flow similar to Swamy's: we first get a half-integral solution (except for a few "bad" facilities), create pairs of half-open facilities, and then open one facility in each pair. By making several improvements, we reduce the approximation ratio to 17.46. The first improvement is a simple modification to the pairing process so that every half-open facility is guaranteed either itself or its closest neighbor to be open (versus having to go through two "jumps" to get to an open facility as in [12]). The second improvement is to randomly sample the half-integral solution and condition on the probability that any given facility is "bad". The algorithm can be derandomized with linear loss in the running time.

The third improvement deals with the bad facilities which inevitabley arise due to the knapsack constraint. All previous algorithms used Kumar's bound from [52] to bound the cost of nearby clients when bad facilities must be closed. However, we show that by using a sparsification technique similar in spirit to - but distinct from - that used in [35], we can focus on a sub-instance in which the connection costs of clients are guaranteed to be evenly distributed throughout the instance. This allows for a much stronger bound than Kumar's, and also results in an LP with bounded integrality gap, unlike previous algorithms.

4.3 Preliminaries

4.3.1 An LP Relaxation

Let $n = |\mathcal{F}| + |\mathcal{C}|$ be the size of the instance. For the ease of analysis, we assume that each client has unit demand. (Note that our algorithm easily extends to the general case.) For any client $j \in \mathcal{C}$, the connection cost of j, denoted as $\text{cost}(j)$, is the distance from j to the nearest *open* facility in our solution. Again, the goal is to *open* a subset $\mathcal{S} \subseteq \mathcal{F}$ of facilities such that the total connection cost is minimized, subject to the *knapsack constraint* $\sum_{i \in \mathcal{S}} w_i \le B$.

The natural Linear Program (LP) relaxation of this problem is as follows.

$$\begin{aligned}
\text{minimize} \quad & \sum_{i \in \mathcal{F}, j \in \mathcal{C}} d(i,j) x_{ij} & \\
\text{subject to} \quad & \sum_{i \in \mathcal{F}} x_{ij} = 1 & \forall j \in \mathcal{C} \\
& x_{ij} \le y_i & \forall i \in \mathcal{F}, j \in \mathcal{C} \\
& \sum_{i \in \mathcal{F}} w_i y_i \le B & \\
& 0 \le x_{ij}, y_i \le 1 & \forall i \in \mathcal{F}, j \in \mathcal{C}
\end{aligned}$$

In this LP, x_{ij} and y_i are indicator variables for the event client j is connected to facility i and facility i is open, respectively. We shall refer to the y_i's as opening variables. The first constraint guarantees that each client is connected to some facility. The second constraint says that client j can only connect to facility i if it is open. The third one is the knapsack constraint.

In this chapter, given a KM instance $\mathcal{I} = (B, \mathcal{F}, \mathcal{C}, d, w)$, let $\mathrm{OPT}_{\mathcal{I}}$ and OPT_f be the cost of the optimal integral solution and the optimal value of the LP relaxation, respectively. Suppose $\mathcal{S} \subseteq \mathcal{F}$ is a solution to $\mathcal{I}$, let $\mathrm{cost}_{\mathcal{I}}(\mathcal{S}) := \sum_{j \in \mathcal{C}} d(j, \mathcal{S})$ denote cost of $\mathcal{S}$, where $d(j, \mathcal{S}) = \min\{d(i,j) : i \in \mathcal{S}\}$. Let (x, y) denote the optimal (fractional) solution of the LP relaxation. Let $C_j := \sum_{i \in \mathcal{F}} d(i,j) x_{ij}$ be the fractional connection cost of j. Given $\mathcal{S} \subseteq \mathcal{F}$ and a vector $v \in \mathbb{R}^{|\mathcal{F}|}$, let $v(\mathcal{S}) := \sum_{i \in \mathcal{S}} v_i$. From now on, let us fix any optimal integral solution of the instance for the analysis.

4.3.2 Kumar's bound

The main technical difficulty of the KM problem is related to the unbounded integrality gap of the LP relaxation. It is known that this gap remains unbounded even when we strengthen the LP with knapsack cover inequalities [51]. All previous constant-factor approximation algorithms for KM rely on Kumar's bound from [52] to get around the gap. Specifically, Kumar's bound is useful to bound the connection cost of a group of clients via some *cluster center* in terms of $\mathrm{OPT}_{\mathcal{I}}$ instead of OPT_f. We now review this bound, and will improve it later.

Lemma 4.3.1. *For each client j, we can compute (in polynomial time) an upper-bound U_j on the connection cost of j in the optimal integral solution (i.e.* $\mathrm{cost}(j) \leq U_j$*) such that*

$$\sum_{j' \in \mathcal{C}} \max\{0, U_j - d(j, j')\} \leq \mathrm{OPT}_{\mathcal{I}}.$$

Proof. We first guess $\mathrm{OPT}_{\mathcal{I}}$ by enumerating the powers of $(1+\epsilon)$ for some small constant $\epsilon > 0$. (We lose a factor of $(1+\epsilon)$ in the approximation ratio and a factor of $O\left(\frac{\log n}{\epsilon}\right)$ in the running time.) Now fix any optimal solution and assume that j connects to i and j' connects to i'. Then, by triangle inequality,

$$\operatorname{cost}(j) = d(i,j) \leq d(i',j) \leq d(j,j') + d(i',j') = d(j,j') + \operatorname{cost}(j'),$$

or equivalently,

$$\operatorname{cost}(j') \geq \operatorname{cost}(j) - d(j,j').$$

Taking the sum over all $j' \neq j$, we have

$$\mathrm{OPT}_{\mathcal{I}} \geq \sum_{j' \neq j} \max\{0, \operatorname{cost}(j) - d(j,j')\}.$$

Then we can simply take U_j such that

$$\mathrm{OPT}_{\mathcal{I}} = \sum_{j' \in \mathcal{C}} \max\{0, U_j - d(j,j')\}.$$

(Observe that the RHS is a linear function of U_j.) □

We can slightly strengthen the LP relaxation by adding the constraints: $x_{ij} = 0$ for all $d(i,j) > U_j$. (Unfortunately, the integrality gap is still unbounded after this step.) Thus we may assume that (x,y) satisfies all these constraints.

Lemma 4.3.2 (Kumar's bound)**.** *Let $\mathcal{S}$ be a set of clients and $s \in \mathcal{S}$, where $d(j,s) \leq$*

βC_j for all $j \in S$ and some constant $\beta \geq 1$, then

$$|\mathcal{S}|U_s \leq \mathrm{OPT}_{\mathcal{I}} + \beta \sum_{j \in \mathcal{S}} C_j.$$

Proof.

$$|\mathcal{S}|U_s = \sum_{j \in \mathcal{S}} U_s = \sum_{j \in \mathcal{S}} (U_s - d(j,s)) + \sum_{j \in \mathcal{S}} d(j,s) \leq \mathrm{OPT}_{\mathcal{I}} + \beta \sum_{j \in \mathcal{S}} C_j,$$

where we use the property of U_s from Lemma 4.3.1 for the last inequality. □

This bound allows us to bound the cost of clients which rely on the bad facility.

4.4 Improved approximation algorithms

4.4.1 Sparse Instances

Kumar's bound can only be tight when the connection cost in the optimal solution is highly concentrated around a single client. However, if this were the case, we could guess the client for which this occurs, along with its optimal facility, which would give us a large advantage. On the other hand, if the connection cost is evenly distributed, we can greatly strengthen Kumar's bound. This is the idea behind our definition of sparse instances below.

Let $\mathrm{CBall}(j,r) := \{k \in \mathcal{C} : d(j,k) \leq r\}$ denote the set of clients within radius of r from client j. Let λ_j be the connection cost of j in the optimal integral solution. Also, let $i(j)$ denote the facility serving j in the optimal solution.

Definition 4.4.1. *Given some constants* $0 < \delta, \epsilon < 1$*, we say that a knapsack median instance* $\mathcal{I} = (B, \mathcal{F}, \mathcal{C}, d, w)$ *is* (δ, ϵ)*-sparse if, for all* $j \in \mathcal{C}$*,*

$$\sum_{k \in \text{CBall}(j, \delta\lambda_j)} (\lambda_j - d(j,k)) \leq \epsilon \text{OPT}_{\mathcal{I}}.$$

We will show that the integrality gap is bounded on these sparse instances. We also give a polynomial-time algorithm to *sparsify* any knapsack median instance. Moreover, the solution of a sparse instance can be used as a solution of the original instance with only a small loss in the total cost.

Lemma 4.4.1. *Given some knapsack median instance* $\mathcal{I}_0 = (B, \mathcal{F}, \mathcal{C}_0, d, w)$ *and* $0 < \delta, \epsilon < 1$*, there is an efficient algorithm that outputs* $O(n^{2/\epsilon})$ *pairs of* $(\mathcal{I}, \mathcal{F}')$*, where* $\mathcal{I} = (B, \mathcal{F}, \mathcal{C}, d, w)$ *is a new instance with* $\mathcal{C} \subseteq \mathcal{C}_0$*, and* $\mathcal{F}' \subseteq \mathcal{F}$ *is a partial solution of* $\mathcal{I}$*, such that at least one of these instances is* (δ, ϵ)*-sparse.*

Proof. Fix any optimal integral solution of $\mathcal{I}_0$. Consider the following algorithm which transforms $\mathcal{I}_0$ into a sparse instance, *assuming for now that we know its optimal solution*:

- Initially, $\mathcal{C} := \mathcal{C}_0$.
- While the instance $(B, \mathcal{F}, \mathcal{C}, d, w)$ is not sparse, i.e. there exists a "bad" client j such that $\sum_{k \in \text{CBall}(j, \delta\lambda_j)} (\lambda_j - d(j,k)) > \epsilon \text{OPT}_{\mathcal{I}}$, *remove* all clients in $\text{CBall}(j, \delta\lambda_j)$ from $\mathcal{C}$.

Note that this algorithm will terminate after at most $1/\epsilon$ iterations: for each $k \in \text{CBall}(j, \delta\lambda_j)$ and its serving facility $i(k)$ in the optimal solution, we have $d(j, i(k)) \leq$

$d(j,k) + \lambda_k$, which implies

$$\sum_{k \in \mathrm{CBall}(j,\delta\lambda_j)} \lambda_k \geq \sum_{k \in \mathrm{CBall}(j,\delta\lambda_j)} (d(j,i(k)) - d(j,k))$$
$$\geq \sum_{k \in \mathrm{CBall}(j,\delta\lambda_j)} (\lambda_j - d(j,k)) > \epsilon \mathrm{OPT}_{\mathcal{I}},$$

and there can be at most $1/\epsilon$ such balls.

Now, while we do not know which client j is "bad" and which facility i serves client j in the optimal solution, we can still "guess" these pairs in $O(n^2)$ time in each iteration. Specifically, we will guess the number of iterations that the above algorithm terminates and the pair $(j, i(j))$ in each iteration. There are at most $O(n^{2/\epsilon})$ possible cases and we will generate all of these new instances. Finally, we include all the facilities $i(j)$ during the process into the set $\mathcal{F}'$ of the corresponding instance. $\square$

The following theorem says that if we have an approximate solution to a sparse instance, then its cost on the original instance is only blown up by a small constant factor.

Theorem 4.4.1. *Let $\mathcal{I} = (B, \mathcal{F}, \mathcal{C}, d, w)$ be a (δ, ϵ)-sparse instance obtained from $\mathcal{I}_0 = (B, \mathcal{F}, \mathcal{C}_0, d, w)$ (by the procedure in the proof of Lemma 4.4.1) and $\mathcal{F}'$ be the corresponding partial solution. If $\mathcal{S} \supseteq \mathcal{F}'$ is any approximate solution to $\mathcal{I}$ (including those open facilities in $\mathcal{F}'$) such that*

$$\mathrm{cost}_{\mathcal{I}}(\mathcal{S}) \leq \alpha \mathrm{OPT}_{\mathcal{I}},$$

then

$$\text{cost}_{\mathcal{I}_0}(\mathcal{S}) \leq \max\left\{\frac{1+\delta}{1-\delta}, \alpha\right\} \text{OPT}_{\mathcal{I}_0}.$$

Proof. For any $k \in \mathcal{C}_0 \backslash \mathcal{C}$, let $\text{CBall}(j, \delta\lambda_j)$ be the ball containing k that was removed from $\mathcal{C}_0$ in the preprocessing phase in Lemma 4.4.1. Recall that $i(j)$ is the facility serving j in the optimal solution. We have

$$\lambda_k \geq \lambda_j - d(j,k) \geq (1-\delta)\lambda_j,$$

which implies,

$$d(k, i(j)) \leq d(j,k) + \lambda_j \leq (1+\delta)\lambda_j \leq \frac{1+\delta}{1-\delta}\lambda_k.$$

Then, by connecting all $k \in \mathcal{C}_0 \setminus \mathcal{C}$ to the corresponding facility $i(j)$ (which is guaranteed to be open because $i(j) \in \mathcal{F}'$), we get

$$\begin{aligned}
\text{cost}_{\mathcal{I}_0}(\mathcal{S}) &= \sum_{k \in \mathcal{C}_0 \backslash \mathcal{C}} \text{cost}(k) + \sum_{k \in \mathcal{C}} \text{cost}(k) \\
&\leq \frac{1+\delta}{1-\delta} \sum_{k \in \mathcal{C}_0 \backslash \mathcal{C}} \lambda_k + \alpha \text{OPT}_{\mathcal{I}} \\
&\leq \frac{1+\delta}{1-\delta} \sum_{k \in \mathcal{C}_0 \backslash \mathcal{C}} \lambda_k + \alpha \sum_{k \in \mathcal{C}} \lambda_k \\
&\leq \max\left\{\frac{1+\delta}{1-\delta}, \alpha\right\} \text{OPT}_{\mathcal{I}_0}.
\end{aligned}$$

□

Note that our notion of sparsity differs from that of Li and Svensson in several ways. It is client-centric, and removes clients instead of facilities from the instance. On the negative side, removed clients' costs blow up by $\frac{1+\delta}{1-\delta}$, so our final approximation cannot guarantee better.

From now on, assume that we are given some arbitrary knapsack median instance $\mathcal{I}_0 = (B, \mathcal{F}, \mathcal{C}_0, d, w)$. We will transform $\mathcal{I}_0$ into a (δ, ϵ)-sparse instance $\mathcal{I}$ and use Theorem 4.4.1 to bound the real cost at the end.

4.4.2 Improving Kumar's bound and modifying the LP relaxation

Here we will show how to improve Kumar's bound in sparse instances. Recall that, for all $j \in \mathcal{C}$, we have

$$\sum_{k \in \mathrm{CBall}(j, \delta\lambda_j)} (\lambda_j - d(j,k)) \leq \epsilon \mathrm{OPT}_{\mathcal{I}}.$$

Then, as before, we can guess $\mathrm{OPT}_{\mathcal{I}}$ and take the maximum U_j such that

$$\sum_{k \in \mathrm{CBall}(j, \delta U_j)} (U_j - d(j,k)) \leq \epsilon \mathrm{OPT}_{\mathcal{I}}.$$

(Observe that the LHS is an increasing function of U_j.) Now the constraints $x_{ij} = 0$ for all $i \in \mathcal{F}, j \in \mathcal{C} : d(i,j) > U_j$ are valid and we can add these into the LP. We also add the following constraints: $y_i = 1$ for all facilities $i \in \mathcal{F}'$. From now on, assume that (x, y) is an optimal solution of this new LP, satisfying all the mentioned constraints.

Lemma 4.4.2. *Let s be any client in sparse instance $\mathcal{I}$ and $\mathcal{S}$ be a set of clients* *such that $d(j,s) \le \beta C_j$ for all $j \in \mathcal{S}$ and some constant $\beta \ge 1$. Then*

$$|\mathcal{S}|U_s \le \epsilon \mathrm{OPT}_{\mathcal{I}} + \frac{\beta}{\delta}\sum_{j\in\mathcal{S}} C_j.$$

Proof. Consider the following two cases.

- For clients $j \in \mathcal{S}' = \mathcal{S} \cap \mathrm{CBall}(s, \delta U_s)$, by definition of sparsity, we have

$$\begin{aligned}|\mathcal{S}'|U_s &= \sum_{j\in\mathcal{S}'}(U_s - d(j,s)) + \sum_{j\in\mathcal{S}'} d(j,s)\\ &\le \epsilon \mathrm{OPT}_{\mathcal{I}} + \beta\sum_{j\in\mathcal{S}'} C_j.\end{aligned}$$

- For clients $j \in \mathcal{S}'' = \mathcal{S} \setminus \mathrm{CBall}(s, \delta U_s)$, we have $\beta C_j \ge d(j,s) \ge \delta U_s$ and we get an alternative bound $U_s \le \frac{\beta}{\delta}C_j$. Thus,

$$|\mathcal{S}''|U_s = \sum_{j\in\mathcal{S}''} U_s \le \sum_{j\in\mathcal{S}''}\frac{\beta}{\delta}C_j.$$

The lemma follows by taking the sum of these two cases. □

4.4.3 Filtering Phase

We will apply the standard filtering method for facility-location problems (see [12, 33]). Basically, we choose a subset $\mathcal{C}' \subseteq \mathcal{C}$ such that clients in $\mathcal{C}'$ are *far* from each other. After assigning each facility to the closest client in $\mathcal{C}'$, it is possible to lower-bound the opening volume of each cluster. Each client in $\mathcal{C}'$ is called a cluster center.

Filtering algorithm: Initialize $\mathcal{C}' := \mathcal{C}$. For each client $j \in \mathcal{C}'$ in increasing order of C_j, we remove all other clients j' such that $d(j, j') \leq 4C_{j'} = 4\max\{C_{j'}, C_j\}$ from $\mathcal{C}'$.

For each $j \in \mathcal{C}'$, define $F_j := \{i \in \mathcal{F} : \ d(i,j) = \min_{k \in \mathcal{C}'} d(i,k)\}$, breaking ties arbitrarily. Let $F'_j = \{i \in F_j : d(i,j) \leq 2C_j\}$ and $\gamma_j = \min_{i \notin F_j} d(i,j)$. Then define $G_j = \{i \in F_j : d(i,j) \leq \gamma_j\}$. We also reassign $y_i := x_{ij}$ for $i \in G_j$ and $y_i := 0$ otherwise. For $j \in \mathcal{C}'$, let M_j be the set containing j and all clients removed by j in the filtering process.

We note that the solution (x, y) may not be feasible to the LP anymore after the reassignment step. For the rest of the paper, we will focus on rounding y into an integral vector. One important property is that the knapsack constraint still holds. In other words, the new sum $\sum_{i \in \mathcal{F}} w_i y_i$ is still at most the budget B. This is due to the fact that $x_{ij} \leq y_i$. The opening variables only decrease after this step; and hence, the knapsack constraint will be preserved.

Lemma 4.4.3. *We have the following properties:*

- *All sets G_j are disjoint,*
- $1/2 \leq y(F'_j)$ *and* $y(G_j) \leq 1$ *for all* $j \in \mathcal{C}'$.
- $F'_j \subseteq G_j$ *for all* $j \in \mathcal{C}'$.

Proof. For the first claim, observe that all F_j's are disjoint and $G_j \subseteq F_j$ by definition. Also, if $\sum_{i \in F'_j} y_i = \sum_{i \in F'_j} x_{ij} < 1/2$, then $\sum_{i \in \mathcal{F} \setminus F'_j} x_{ij} > 1/2$. Since the radius of F'_j is $2C_j$, this means that $C_j > (1/2)(2C_j) = C_j$, which is a contradiction. Since we reassign $y_i := x_{ij}$ for all $i \in G_j$, the volume $y(G_j)$ is now at most 1. Finally, we have $2C_j \leq \gamma_j$. Otherwise, let $i \notin F_j$ be the facility such that $\gamma_j = d(i,j)$. Observe that facility i is claimed by another cluster center, say j', because $d(i,j') \leq d(i,j) \leq 2C_j$. This implies that $d(j,j') \leq d(i,j) + d(i,j') \leq 4C_j$, which is a contradiction. □

It is clear that for all $j, j' \in \mathcal{C}'$, $d(j,j') \geq 4\max\{C_{j'}, C_j\}$. Moreover, for each $j \in \mathcal{C} \setminus \mathcal{C}'$, we can find $j' \in \mathcal{C}'$, where j' causes the removal of j, or, in other words, $C_{j'} \leq C_j$ and $d(j,j') \leq 4C_j$. Assuming that we have a solution $\mathcal{S}$ for the instance $\mathcal{I}' = (B, \mathcal{F}, \mathcal{C}', d, w)$ where each client j in $\mathcal{C}'$ has demand $m_j = |M_j|$ (i.e. there are $|M_j|$ copies of j), we can transform it into a solution for $\mathcal{I}$ as follows. Each client $j \in \mathcal{C} \setminus \mathcal{C}'$ will be served by the facility of j' that removed j. Then $\text{cost}(j) = d(j,j') + \text{cost}(j') \leq \text{cost}(j') + 4C_j$. Therefore,

$$\begin{aligned}
\text{cost}_{\mathcal{I}}(\mathcal{S}) &= \sum_{j \in \mathcal{C}'} \text{cost}(j) + \sum_{j \in \mathcal{C} \setminus \mathcal{C}'} \text{cost}(j) \\
&\leq \sum_{j \in \mathcal{C}'} \text{cost}(j) + \sum_{j \in \mathcal{C} \setminus \mathcal{C}'} (\text{cost}(j'(j)) + 4C_j) \\
&\leq \text{cost}_{\mathcal{I}'}(\mathcal{S}) + 4\text{OPT}_f.
\end{aligned}$$

where, in the second line, $j'(j)$ is the center in $\mathcal{C}'$ that removed j.

4.4.4 A $(23.09 + \epsilon)$-approximation algorithm

In this section, we describe a simple randomized $(23.09 + \epsilon)$-approximation algorithm. In the next section, we will derandomize it and give more insights to further improve the approximation ratio to $17.46 + \epsilon$.

High-level ideas: We reuse Swamy's idea from [12] to first obtain an *almost* half integral solution $\hat{y}$. This solution $\hat{y}$ has a very nice structure. For example, each client j only (fractionally) connects to at most 2 facilities, and there is at least a half-opened facility in each G_j. We shall refer to this set of 2 facilities as a bundle. In [12], the author applies a standard clustering process to get disjoint bundles and round $\hat{y}$ by opening at least one facility per bundle. The drawback of this method is that we have to pay extra cost for bundles removed in the clustering step. In fact, it is possible to open at least one facility per bundle without filtering out any bundle. The idea here is inspired by the work of Charikar et. al [33]. In addition, instead of picking $\hat{y}$ deterministically, sampling such a half integral extreme point will be very helpful for the analysis.

We consider the following polytope.

$$\mathcal{P} = \{v \in [0,1]^{|\mathcal{F}|} : v(F'_j) \geq 1/2,\ v(G_j) \leq 1,\ \forall j \in \mathcal{C}';\quad \sum_{i \in \mathcal{F}} w_i v_i \leq B\}.$$

Lemma 4.4.4 ([12])**.** *Any extreme point of $\mathcal{P}$ is almost half-integral: there exists at most 1 cluster center $s \in \mathcal{C}'$ such that G_s contains variables $\notin \{0, \frac{1}{2}, 1\}$. We call s a fractional client.*

Notice by Lemma 4.4.3 that $y \in \mathcal{P}$. By Carathéodory's theorem, y is a convex combination of at most $t = |\mathcal{F}| + 1$ extreme points of $\mathcal{P}$. Moreover, there is an efficient algorithm based on the ellipsoid method to find such a decomposition (e.g., see [88]). We apply this algorithm to get extreme points $y^{(1)}, y^{(2)}, \ldots, y^{(t)} \in \mathcal{P}$ and

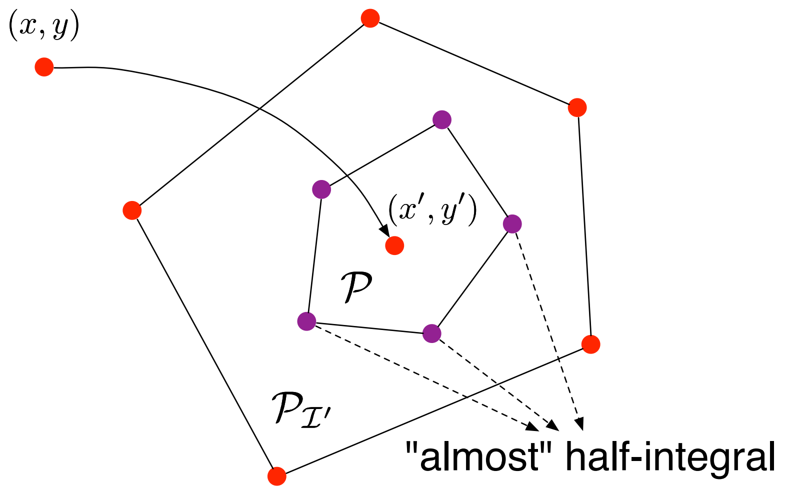

Figure 4.1: After the filtering step, the LP solution lie in polytope $\mathcal{P}$ which has "almost" half-integral extreme points.

coefficients $0 \le p_1, \ldots, p_t \le 1, \sum_{i=1}^{t} p_i = 1$, such that

$$y = p_1 y^{(1)} + p_2 y^{(2)} + \ldots + p_t y^{(t)}.$$

This representation defines a distribution on t extreme points of $\mathcal{P}$. Let $Y \in [0,1]^{\mathcal{F}}$ be a random vector where $\Pr[Y = y^{(i)}] = p_i$ for $i = 1, \ldots, t$. Observe that Y is *almost* half-integral. Let s be the fractional client in Y. (We assume that s exists; otherwise, the cost will only be smaller.)

Defining primary and secondary facilities: For each $j \in \mathcal{C}'$,

- If $j \neq s$, let $i_1(j)$ be any half-integral facility in F'_j (i.e. $Y_{i_1(j)} = 1/2$; such a facility exists because $Y(F'_j) \ge 1/2$). Else ($j = s$), let $i_1(j)$ be the smallest-weight facility in F'_j with $Y_{i_1(j)} > 0$.

- If $Y(i_1(j)) = 1$, let $i_2(j) = i_1(j)$.

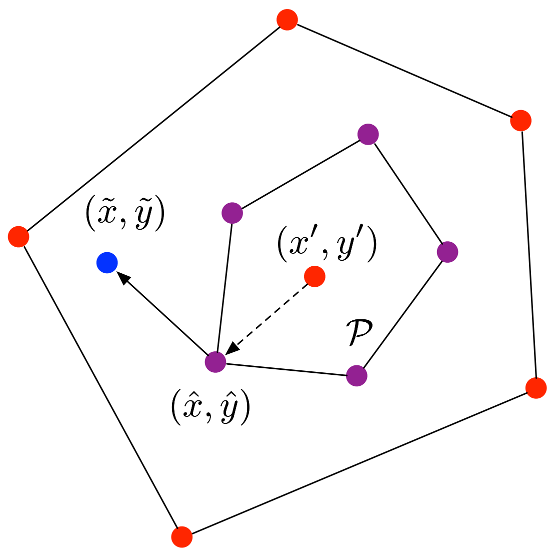

Figure 4.2: Our strategy is to randomly round the LP solution into one vertex $(\hat{x}, \hat{y})$ of $\mathcal{P}$ and then round it into an integral solution $(\tilde{x}, \tilde{y})$.

- If $Y(G_j) < 1$, then let $\sigma(j)$ be the nearest client to j in $\mathcal{C}'$. Define $i_2(j) = i_1(\sigma(j))$.
- If $Y(G_j) = 1$, then
  - If $j \neq s$, let $i_2(j)$ be the other half-integral facility in G_j.
  - Else ($j = s$), let $i_2(j)$ be the smallest-weight facility in G_j with $Y_{i_2(j)} > 0$. If there are ties and $i_1(j)$ is among these facilities then we let $i_2(j) = i_1(j)$.
- We call $i_1(j), i_2(j)$ the primary facility and the secondary facility of j, respectively.

Constructing the neighborhood graph: Initially, construct the directed graph $\mathcal{G}$ on clients in $\mathcal{C}'$ such that there is an edge $j \to \sigma(j)$ for each $j \in \mathcal{C}' : Y(G_j) < 1$. Note that all vertices in $\mathcal{G}$ have outdegree ≤ 1. If $Y(G_j) = 1$, then vertex j has no outgoing edge. In this case, we replace j by the edge $i_1(j) \to i_2(j)$, instead. Finally,

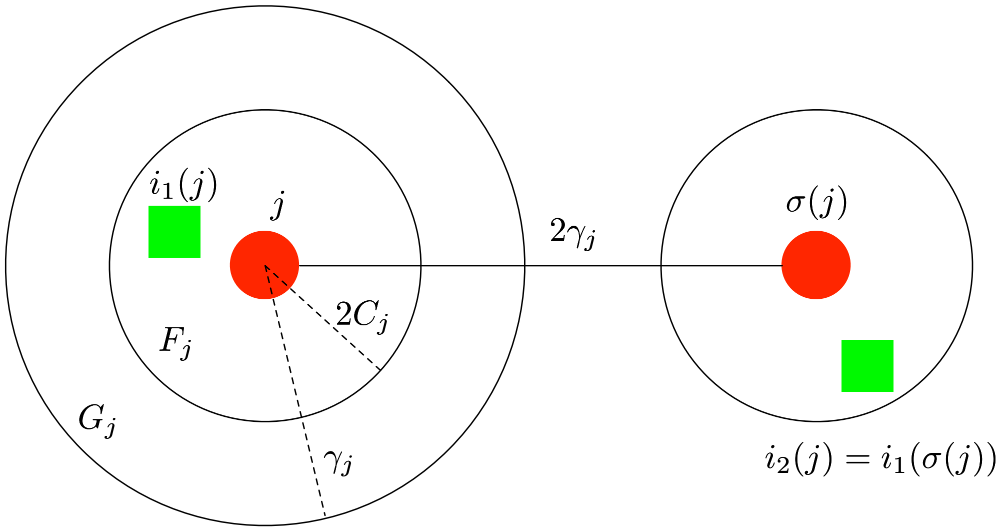

Figure 4.3: Illustration of $F'_j, G_j, i_1(j), i_2(j)$, and $\sigma(j)$.

we relabel all other nodes in $\mathcal{G}$ by its primary facility. Now we can think of each client $j \in \mathcal{C}'$ as an edge from $i_1(j)$ to $i_2(j)$ in $\mathcal{G}$.

Lemma 4.4.5. *Without loss of generality, we can assume that all cycles of $\mathcal{G}$ (if any) are of size 2. This means that $\mathcal{G}$ is bipartite.*

Proof. Since the maximum outdegree is equal to 1, each (weakly) connected component of $\mathcal{G}$ has at most 1 cycle. Consider any cycle $j \to \sigma(j) \to \sigma^2(j) \to \ldots \to \sigma^k(j) \to j$. Then it is easy to see that $d(j, \sigma(j)) = d(\sigma^k(j), j)$. The argument holds for any j in the cycle, and all edges on the cycle have the same length. Then we can simply redefine $\sigma(\sigma^k(j)) := \sigma^{k-1}(j)$ and get a cycle of size 2 instead. We can also change the secondary of the client corresponding to the edge $(\sigma^k(j), j)$ into $\sigma^{k-1}(j)$ because they are both at the same distance from it. □

We are now ready to describe the main algorithm.

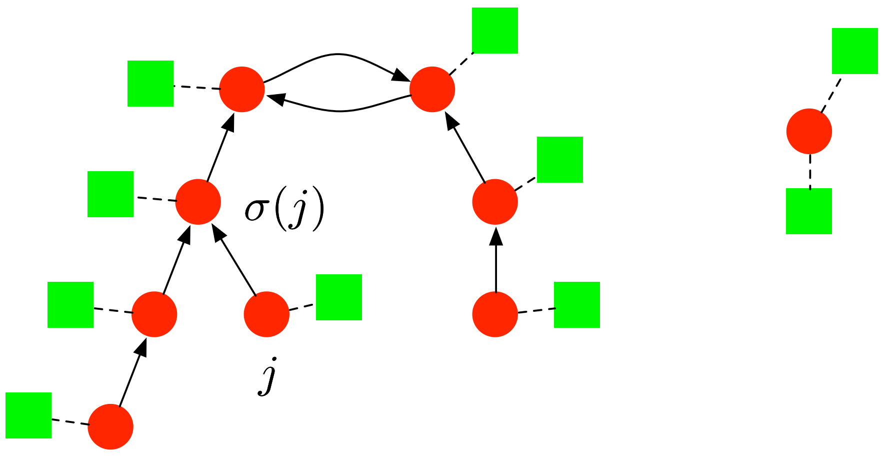

Figure 4.4: Illustration of a neighborhood graph. All cycles are of size two.

Algorithm 10 Round(Y)

```
1: Construct the neighborhood graph $\mathcal{G}$ based on $Y$
2: Let $C_1, C_2$ be independent sets which partition $\mathcal{G}$
3: Let $W_1, W_2$ be the total weight of the facilities in $C_1, C_2$ respectively.
4: if $W_1 \leq W_2$ then
5:     return $C_1$
6: else
7:     return $C_2$
```

Theorem 4.4.2. *Algorithm 11 returns a feasible solution $\mathcal{S}$ where*

$$\mathbf{E}[\text{cost}_{\mathcal{I}_0}(\mathcal{S})] \leq \max\left\{\frac{1+\delta}{1-\delta}, 10 + 12/\delta + 3\epsilon\right\} \text{OPT}_{\mathcal{I}_0}.$$

In particular, the approximation ratio is at most $(23.087 + 3\epsilon)$ *when setting* $\delta :=$ 0.916966.

Proof. Assume $\mathcal{I}$ is the sparse instance obtained from $\mathcal{I}_0$. We will give a proof of feasibility and a cost analysis. Recall that s is the center where we may have fractional values Y_i with $i \in G_s$.

Feasibility:

Algorithm 11 BasicAlgorithm(δ, ϵ, $\mathcal{I}_0$)

```
1: Generate $O(n^{2/\epsilon})$ pairs $(\mathcal{I}, \mathcal{F}')$ using the algorithm in the proof of Lemma 4.4.1
2: $\mathcal{S} \leftarrow \emptyset$
3: for each pair $(\mathcal{I}, \mathcal{F}')$ do
4:     Let $(x, y)$ be the optimal solution of the modified LP relaxation in Section 4.4.2.
5:     Apply the filtering algorithm to get $\mathcal{I}'$.
6:     Use $\mathcal{F}, \mathcal{C}'$ to define the polytope $\mathcal{P}$.
7:     Sample a random extreme point $Y$ of $\mathcal{P}$ as described above.
8:     Let $\mathcal{S}' \leftarrow$Round$(Y)$
9:     If $\mathcal{S}'$ is feasible and its cost is smaller than the cost of $\mathcal{S}$ then $\mathcal{S} \leftarrow \mathcal{S}'$.
10: return $\mathcal{S}$
```

- For all centers $j \in \mathcal{C}'$ with $Y(G_j) < 1$, we have

$$w_{i_1(j)} \leq 2 \sum_{i \in G_j} Y_i w_i.$$

  Note that this is true for $j \neq s$ because $Y_{i_1(j)} = 1/2$. Otherwise, $j = s$, by definition, $w_{i_1(j)}$ is the smallest weight in the set F'_s which has volume at least $1/2$. Thus, $w_{i_1(j)} \leq 2 \sum_{i \in F'_s} Y_i w_i \leq 2 \sum_{i \in G_j} Y_i w_i$.

- For all centers $j \in \mathcal{C}'$ with $Y(G_j) = 1$, we have

$$w_{i_1(j)} + w_{i_2(j)} \leq 2 \sum_{i \in G_j} Y_i w_i.$$

  The equality happens when $j \neq s$. Otherwise, $j = s$, we consider the following 2 cases

  - If $i_1(s) = i_2(s)$ the inequality follows because $w_{i_1(j)} = w_{i_2(j)} \leq \sum_{i \in G_j} Y_i w_i$.
  - Else, we have $i_2(s) \in G_j \setminus F'_j$ by definition of $i_2(s)$. Since $w_{i_1(s)} \geq w_{i_2(s)}$

and $Y(F'_s) \geq 1/2$,

$$\begin{aligned}\frac{1}{2}w_{i_1(s)} + \frac{1}{2}w_{i_2(s)} &\leq Y(F'_s)w_{i_1(s)} + (1 - Y(F'_s))w_{i_2(s)} \\ &\leq \sum_{i \in F'_j} Y_i w_i + \sum_{i \in G_j \setminus F'_j} Y_i w_i = \sum_{i \in G_j} Y_i w_i.\end{aligned}$$

Recall that each center $j \in \mathcal{C}'$ is accounted for either one vertex $i_1(j)$ of $\mathcal{G}$ if $Y(G_j) < 1$ or two vertices $i_1(j), i_2(j)$ of $\mathcal{G}$ if $Y(G_j) = 1$). Thus, the total weight of all vertices in $\mathcal{G}$ is at most

$$2 \sum_{j \in \mathcal{C}'} \sum_{i \in G_j} Y_i w_i \leq 2B,$$

where the last inequality follows because $Y \in \mathcal{P}$. It means that either W_1 or W_2 is less than or equal to B, and Algorithm 10 always returns a feasible solution.

Cost analysis:

We show that the expected cost of j can be bounded in terms of γ_j, U_j, and y. For $j \in \mathcal{C}$, let $j'(j)$ denote the cluster center of j and define $j'(j) = j$ if $j \in \mathcal{C}'$. Recall that in the instance $\mathcal{I}' = (B, \mathcal{F}, \mathcal{C}', d)$, each client $j \in \mathcal{C}'$ has demand $m_j = |M_j|$. Notice that

$$\begin{aligned}\mathrm{OPT}_f = \sum_{j \in \mathcal{C}} C_j \geq \sum_{j \in \mathcal{C}} C_{j'(j)} &= \sum_{j \in \mathcal{C}'} m_j C_j \\ &= \sum_{j \in \mathcal{C}'} m_j \left(\sum_{i \in G_j} x_{ij} d(i,j) + \sum_{i \in \mathcal{F} \setminus G_j} x_{ij} d(i,j) \right) \\ &\geq \sum_{j \in \mathcal{C}'} m_j \left(\sum_{i \in G_j} y_i d(i,j) + \gamma_j (1 - y(G_j)) \right). \qquad (4.1)\end{aligned}$$

The last inequality follows because, for any center j, $\sum_{i\in\mathcal{F}} x_{ij} = 1$, and γ_j is the radius of the ball G_j by definition. Now, for $v \in [0,1]^{\mathcal{F}}$, we define

$$B_j(v) := m_j \left(\sum_{i\in G_j} v_i d(i,j) + \gamma_j(1 - v(G_j)) \right).$$

Let $K(v) = \sum_{j\in\mathcal{C}'} B_j(v)$. Recall that $\mathbf{E}[Y_i] = y_i$ for all $i \in \mathcal{F}$. By (4.1) and linearity of expectation, we have

$$\mathbf{E}[K(Y)] = K(y) \le \mathrm{OPT}_f.$$

Also note that

$$\sum_{j\in\mathcal{C}'} \mathbf{E}[B_j(Y)] = \sum_{j\in\mathcal{C}'} B_j(y) \le \sum_{j\in\mathcal{C}'} m_j C_j \le \sum_{j\in\mathcal{C}'} \sum_{k\in M_j} C_k = \sum_{j\in\mathcal{C}} C_j.$$

Next, we will analyze $\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})$. To this end, we shall bound the connection cost of a client j in terms of $B_j(Y)$. Algorithm 10 guarantees that, for each $j \in \mathcal{C}'$, either $i_1(j)$ or $i_2(j)$ is in $\mathcal{S}$. By construction, $d(i_1(j), j) \le d(i_2(j), j)$. In the worst case, we may need to connect j to $i_2(j)$, and hence $\mathrm{cost}(j) \le d_j d(i_2(j), j)$ for all client j.

Fix any client j with $Y(G_j) < 1$. Recall that $\gamma_j = \min_{i\notin F_j} d(i,j)$ and $\sigma(j)$ is the closest client to j in $\mathcal{C}'$. Suppose $\gamma_j = d(i',j)$ where $i' \in F_{j'}$ for some $j' \in \mathcal{C}'$. By definition, $d(i',j') \le \gamma_j$. Then $d(j,\sigma(j)) \le d(j,j') \le d(i',j) + d(i',j') \le 2\gamma_j$. Also, since $i_1(\sigma(j)) \in F'_{\sigma(j)}$, we have that $d(\sigma(j), i_1(\sigma(j))) \le 2C_{\sigma(j)}$. In addition, recall that $4\max\{C_j, C_{\sigma(j)}\} \le d(j,\sigma(j)) \le 2\gamma_j$. Thus, $2C_{\sigma(j)} \le \gamma_j$. Then the following bound holds when $Y(G_j) < 1$:

$$\begin{aligned}
\mathrm{cost}(j) &\leq m_j d(i_2(j), j) \\
&\leq m_j(d(j, \sigma(j)) + d(\sigma(j), i_2(j))) \\
&= m_j(d(j, \sigma(j)) + d(\sigma(j), i_1(\sigma(j)))) \\
&\leq m_j(2\gamma_j + 2C_{\sigma(j)}) \\
&\leq 3m_j\gamma_j.
\end{aligned}$$

Consider the following cases.

- If $j \neq s$, then either $Y(G_j) = 1$ or $Y(G_j) = 1/2$.
  - Case $Y(G_j) = 1$: then $Y_{i_1(j)} = Y_{i_2(j)} = 1/2$, we have

    $$\mathrm{cost}(j) \leq m_j d(i_2(j), j) \leq 2m_j \sum_{i \in G_j} Y_i d(i, j) = 2B_j(Y).$$

  - Case $Y(G_j) = 1/2$: we have

    $$\mathrm{cost}(j) \leq 3m_j\gamma_j = 6m_j\gamma_j(1 - Y(G_j)) \leq 6B_j(Y).$$

- If $j = s$, we cannot bound the cost in terms of $B_j(Y)$. Instead, we shall use Kumar's bound.
  - Case $Y(G_j) = 1$: $i_2(j) \in G_j$. Recall that U_j is the upper-bound on the connection cost of j. Our LP constraints guarantee that $x_{ij} = 0$ for all

$d(i,j) > U_j$. Since $Y_{i_2(j)} > 0$, we also have $y_{i_2(j)} > 0$ or $x_{i_2(j)j} > 0$, which implies that $d(i_2(j), j) \leq U_j$. Thus,

$$\mathrm{cost}(j) \leq m_j d(i_2(j), j) \leq m_j U_j.$$

– Case $Y(G_j) < 1$: then there must exists some facility $i \notin G_j$ such that $x_{ij} > 0$. Since γ_j is the radius of G_j, we have $\gamma_j \leq d(i,j) \leq U_j$; and hence,

$$\mathrm{cost}(j) \leq 3m_j\gamma_j \leq 3m_jU_j.$$

In either cases, applying the improved Kumar's bound to the cluster M_s where $d(k,s) \leq 4C_k$ for all $k \in M_s$, we get

$$\begin{aligned}\mathrm{cost}(j) &\leq 3m_jU_j \\ &\leq 3\epsilon\mathrm{OPT}_{\mathcal{I}} + \frac{3 \cdot 4}{\delta}\sum_{k \in M_s} C_k \\ &\leq 3\epsilon\mathrm{OPT}_{\mathcal{I}} + \frac{12}{\delta}\mathrm{OPT}_f.\end{aligned}$$

Now, we will bound the facility-opening cost. Notice that, for all facilities $i \in C_1 \cup C_2$ but at most two facilities $i_1(s)$ and $i_2(s)$, we have $Y_i \in \{1/2, 1\}$.

Then,

$$\begin{aligned}\mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})] &\le \sum_{j\in\mathcal{C}':j\neq s} 6\mathbf{E}[B_j(Y)] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)\mathrm{OPT}_f \\ &= 6\sum_{j\in\mathcal{C}'} B_j(y) + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)\mathrm{OPT}_f \\ &\le 6K(y) + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)\mathrm{OPT}_f \\ &\le (6+12/\delta)\mathrm{OPT}_f + 3\epsilon\mathrm{OPT}_{\mathcal{I}}.\end{aligned}$$

Therefore,

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})] \le \mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})] + 4\mathrm{OPT}_f \le (10+12/\delta)\mathrm{OPT}_f + 3\epsilon\mathrm{OPT}_{\mathcal{I}}.$$

Finally, applying Theorem 4.4.1 to $\mathcal{S}$,

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}_0}(\mathcal{S})] \le \max\left\{\frac{1+\delta}{1-\delta}, 10+12/\delta+3\epsilon\right\}\mathrm{OPT}_{\mathcal{I}_0}.$$

□

4.4.5 A $(17.46+\epsilon)$-approximation algorithm via conditioning on the fractional cluster center

Recall that the improved Kumar's bound for the fractional client s is

$$|M_s|U_s \le \epsilon\mathrm{OPT}_{\mathcal{I}} + (4/\delta)\sum_{j\in M_s} C_j.$$

In Theorem 4.4.2, we upper-bound the term $\sum_{j \in M_s} C_j$ by OPT_f. However, if this is tight, then the fractional cost of all other clients not in M_s must be zero and we should get an improved ratio.

To formalize this idea, let $u \in \mathcal{C}'$ be the client such that $\sum_{j \in M_u} C_j$ is maximum. Let $\alpha \in [0,1]$ such that $\sum_{j \in M_u} C_j = \alpha \mathrm{OPT}_f$, then

$$|M_s| U_s \leq \epsilon \mathrm{OPT}_{\mathcal{I}} + (4/\delta)\alpha \mathrm{OPT}_f. \tag{4.2}$$

The following bound follows immediately by replacing the Kumar's bound by (4.2) in the proof of Theorem 4.4.2.

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})] \leq (10 + 12\alpha/\delta + 3\epsilon)\mathrm{OPT}_{\mathcal{I}}. \tag{4.3}$$

In fact, this bound is only tight when u happens to be the fractional client after sampling Y. If u is not "fractional", the second term in the RHS of (4.2) should be at most $(1-\alpha)\mathrm{OPT}_f$. Indeed, if u is *rarely* a fractional client, we should obtain a strictly better bound. To this end, let $\mathcal{E}$ be the event that u is the fractional client after the sampling phase. Let $p = \Pr[\mathcal{E}]$. We get the following lemma.

Lemma 4.4.6. *Algorithm 11 returns a solution $\mathcal{S}$ with*

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})] \leq (10 + \min\{12\alpha/\delta, (12/\delta)(p\alpha + (1-p)(1-\alpha))\} + 3\epsilon)\mathrm{OPT}_{\mathcal{I}}.$$

Proof. We reuse the notations and the connection cost analysis in the proof of

Theorem 4.4.2. Recall that $\mathcal{E}$ is the event that u is the fractional client. We have

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})|\mathcal{E}] \le 6 \sum_{j\in\mathcal{C}':j\neq u} \mathbf{E}[B_j(Y)|\mathcal{E}] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12\alpha/\delta)\mathrm{OPT}_f.$$

If $\bar{\mathcal{E}}$ happens, assume $s \neq u$ is the fractional one and let $\bar{\mathcal{E}}(s)$ denote this event. Then,

$$\begin{aligned}\mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})|\bar{\mathcal{E}}(s)] &\le 6 \sum_{j\in\mathcal{C}':j\neq s} \mathbf{E}[B_j(Y)|\bar{\mathcal{E}}(s)] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)(1-\alpha)\mathrm{OPT}_f \\ &\le 6 \sum_{j\in\mathcal{C}'} \mathbf{E}[B_j(Y)|\bar{\mathcal{E}}(s)] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)(1-\alpha)\mathrm{OPT}_f\end{aligned}$$

Therefore,

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})|\bar{\mathcal{E}}] \le 6 \sum_{j\in\mathcal{C}'} \mathbf{E}[B_j(Y)|\bar{\mathcal{E}}] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)(1-\alpha)\mathrm{OPT}_f.$$

Also, $(1-p)\mathbf{E}[B_u(Y)|\bar{\mathcal{E}}] \le \mathbf{E}[B_u(Y)]$ because $B_u(Y)$ is always non-negative. The total expected cost can be bounded as follows.

$$\begin{aligned}
\mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})] &= p\mathbf{E}[cost_{\mathcal{I}'}(\mathcal{S})|\mathcal{E}] + (1-p)\mathbf{E}[cost_{\mathcal{I}'}(\mathcal{S})|\bar{\mathcal{E}}] \\
&\leq 6 \sum_{j \in \mathcal{C}' : j \neq u} \mathbf{E}[B_j(Y)] + 3\epsilon \mathrm{OPT}_{\mathcal{I}} \\
&\qquad + (12/\delta)(p\alpha + (1-p)(1-\alpha))\mathrm{OPT}_f + 6(1-p)\mathbf{E}[B_u(Y)|\bar{\mathcal{E}}] \\
&\leq 6 \sum_{j \in \mathcal{C}'} \mathbf{E}[B_j(Y)] + 3\epsilon \mathrm{OPT}_{\mathcal{I}} + (12/\delta)(p\alpha + (1-p)(1-\alpha))\mathrm{OPT}_f. \\
&\leq 6K(y) + (3\epsilon + (12/\delta)(p\alpha + (1-p)(1-\alpha)))\mathrm{OPT}_{\mathcal{I}} \\
&\leq (6 + 3\epsilon + (12/\delta)(p\alpha + (1-p)(1-\alpha)))\mathrm{OPT}_{\mathcal{I}}. && (4.4)
\end{aligned}$$

The lemma follows due to (4.3), (4.4), and the fact that $\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})] \leq \mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})] + 4\mathrm{OPT}_f$. □

Finally, conditioning on the event $\mathcal{E}$, we are able to combine certain terms and get the following improved bound.

Lemma 4.4.7. *Algorithm 11 returns a solution $\mathcal{S}$ with*

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})|\mathcal{E}] \leq (\max\{6/p, 12/\delta\} + 4 + 3\epsilon)\mathrm{OPT}_{\mathcal{I}}.$$

Proof. Again, since $B_j(Y) \geq 0$ for all $j \in \mathcal{C}'$ and all Y, we have $\mathbf{E}[B_j(Y)|\mathcal{E}] \leq \mathbf{E}[B_j(Y)]/p$. Also, recall that $\mathbf{E}[B_j(Y)] = B_j(y) \leq d_j C_j \leq \sum_{k \in M_j} C_k$ for any $j \in \mathcal{C}'$.

Therefore,

$$\begin{aligned}
\mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})|\mathcal{E}] &\le 6 \sum_{j\in\mathcal{C}':j\neq u} \mathbf{E}[B_j(Y)|\mathcal{E}] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)\sum_{j\in M_u} C_j \\
&\le (6/p) \sum_{j\in\mathcal{C}':j\neq u} \mathbf{E}[B_j(Y)] + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)\sum_{j\in M_u} C_j \\
&\le (6/p) \sum_{j\in\mathcal{C}:j\notin M_u} C_j + 3\epsilon\mathrm{OPT}_{\mathcal{I}} + (12/\delta)\sum_{j\in M_u} C_j \\
&\le \max\{6/p, 12/\delta\}\sum_{j\in\mathcal{C}} C_j + 3\epsilon\mathrm{OPT}_{\mathcal{I}} \\
&\le \max\{6/p, 12/\delta\}\mathrm{OPT}_f + 3\epsilon\mathrm{OPT}_{\mathcal{I}} \\
&\le (\max\{6/p, 12/\delta\} + 3\epsilon)\mathrm{OPT}_{\mathcal{I}}.
\end{aligned}$$

The lemma follows since $\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})|\mathcal{E}] \le \mathbf{E}[\mathrm{cost}_{\mathcal{I}'}(\mathcal{S})|\mathcal{E}] + 4\mathrm{OPT}_f$. □

Now we have all the required ingredients to get an improved approximation ratio. Algorithm 12 is a derandomized version of Algorithm 11.

Algorithm 12 DETERMINISTICALGORITHM(δ, ϵ, $\mathcal{I}_0$)

```
1: Generate $O(n^{2/\epsilon})$ pairs $(\mathcal{I}, \mathcal{F}')$ using the algorithm in the proof of Lemma 4.4.1
2: $\mathcal{S} \leftarrow \emptyset$
3: for each pair $(\mathcal{I}, \mathcal{F}')$ do
4:     Let $(x, y)$ be the optimal solution of the modified LP relaxation in Section 4.4.2.
5:     Apply the filtering algorithm to get $\mathcal{I}'$.
6:     Use $\mathcal{F}, \mathcal{C}'$ to define the polytope $\mathcal{P}$.
7:     Decompose $y$ into a convex combination of extreme points $y^{(1)}, y^{(2)}, \dots, y^{(t)}$ of $\mathcal{P}$.
8:     for each $Y \in \{y^{(1)}, y^{(2)}, \dots, y^{(t)}\}$ do
9:         Let $\mathcal{S}' \leftarrow$ROUND$(Y)$
10:        If $\mathcal{S}'$ is feasible and its cost is smaller than the cost of $\mathcal{S}$ then $\mathcal{S} \leftarrow \mathcal{S}'$.
11: return $\mathcal{S}$
```

Theorem 4.4.3. *Algorithm 12 returns a feasible solution $\mathcal{S}$ where*

$$\mathrm{cost}_{\mathcal{I}_0}(\mathcal{S}) \leq (17.46 + 3\epsilon)\mathrm{OPT}_{\mathcal{I}_0},$$

when setting $\delta = 0.891647$.

Proof. Again, suppose $\mathcal{I}$ is a sparse instance obtained from $\mathcal{I}_0$. Recall that $p = \Pr[\mathcal{E}]$ is the probability that u, the cluster center with maximum fractional cost $\sum_{j \in M_u} C_j = \alpha \mathrm{OPT}_f$, is fractional. Consider the following cases:

- Case $p \leq 1/2$: By Lemma 4.4.6 and the fact that Algorithm 12 always returns a solution $\mathcal{S}$ from the same distribution with minimum cost, we have

  $$\mathrm{cost}_{\mathcal{I}}(\mathcal{S}) \leq (10 + \min\{12\alpha/\delta, (12/\delta)(p\alpha + (1-p)(1-\alpha))\} + 3\epsilon)\mathrm{OPT}_{\mathcal{I}}.$$

  By Theorem 4.4.1, the approximation ratio is at most

  $$\max\left\{\frac{1+\delta}{1-\delta}, 10 + 3\epsilon + \min\{12\alpha/\delta, (12/\delta)(p\alpha + (1-p)(1-\alpha))\}\right\}.$$

  - If $\alpha \leq 1/2$, the ratio is at most $\max\left\{\frac{1+\delta}{1-\delta}, 10 + 3\epsilon + 6/\delta\right\}$.
  - If $\alpha \geq 1/2$, we have

    $$(12/\delta)(p\alpha + (1-p)(1-\alpha)) = (12/\delta)(p(2\alpha - 1) - \alpha + 1) \leq 6/\delta.$$

    Again, the ratio is at most $\max\left\{\frac{1+\delta}{1-\delta}, 10 + 3\epsilon + 6/\delta\right\}$.

- Case $p \geq 1/2$: Observe that the event $\mathcal{E}$ does happen for some point in the for loop at lines 8, 9, and 10. By Lemma 4.4.7 and the fact that $1+2/p = 3 < 12/\delta$, we have

$$\text{cost}_{\mathcal{I}}(\mathcal{S}) \leq (\max\{6/p, 12/\delta\} + 4 + 3\epsilon)\text{OPT}_{\mathcal{I}} = (12/\delta + 4 + 3\epsilon)\text{OPT}_{\mathcal{I}}.$$

  By Theorem 4.4.1, the approximation ratio is bounded by $\max\left\{\frac{1+\delta}{1-\delta}, \frac{12}{\delta} + 3\epsilon + 4\right\}$.

In all cases, the approximation ratio is at most

$$\max\left\{\frac{1+\delta}{1-\delta}, 12/\delta + 3\epsilon + 4, 10 + 3\epsilon + 6/\delta\right\} \leq 17.4582 + 3\epsilon,$$

when $\delta = 0.89167$. □

Note that in [12], Swamy considered a slightly more general version of KM where each facility also has an opening cost. It can be shown that Theorem 4.4.3 also extends to this variant.

4.5 An improved bi-factor approximation algorithm

In this section, we show that one can obtain a *bi-factor* $(1+\sqrt{3}+\epsilon)$-approximation algorithm for the KM problem if allowed to slightly violate the budget constraint by a factor of $(1+\epsilon)$. This algorithm is inspired by the work of Li-Svenssson [35], in which they gave a $(1+\sqrt{3}+\epsilon)$-approximation algorithm for the k-median problem. The idea is to compute a "bi-point" solution which is a convex combination of a feasible solution and another pseudo solution. Then rounding this solution will result in a solution whose cost can be bounded by $(1+\sqrt{3}+\epsilon)$ times the optimal cost but would slightly violate the knapsack constraint. The rounding step can be done using SRDR and some $O(1)$ left-over fractional variables will have to be rounded up to 1, resulting in a small violation in the total weight. In [35], the authors use a postprocessing step to correct the solution, making the cardinality constraint to be preserved exactly. However, this step does not seem to work for the knapsack median problem. We conjecture that the 2.675-approximation algorithm for k-median in [36] also extends to a bi-factor 2.675-approximation algorithm for KM.

4.5.1 Pruning "big" facilities and computing a bi-point solution

Let $\mathcal{I}_0 = (B_0, \mathcal{F}_0, \mathcal{C}, d, w^{(0)})$ be any KM instance and $\epsilon > 0$ be some small parameter. Let us also fix an optimal integral solution $\mathcal{S}_0$ of $\mathcal{I}_0$. Note that $\mathcal{S}_0$ may contain at most $1/\epsilon$ facilities which have weight greater than or equal to ϵB_0. Let $\mathcal{F}_\epsilon = \{i \in \mathcal{F}_0 : w_i^{(0)} \geq \epsilon B_0\}$ and $\mathcal{S}_\epsilon = \mathcal{S}_0 \cap \mathcal{F}_\epsilon$. Then $|\mathcal{S}_\epsilon| \leq 1/\epsilon$. Hence one may "guess" this set $\mathcal{S}_\epsilon$ in $n^{O(1/\epsilon)}$ time.

From now on, suppose that we already have the "correct" set $\mathcal{S}_\epsilon$. Let $\mathcal{I} = (B, \mathcal{F}, \mathcal{C}, d, w)$ denote the residual instance in which (i) we eliminate all "big" facilities: $\mathcal{F} = (\mathcal{F}_0 \setminus \mathcal{F}_\epsilon) \cup \mathcal{S}_\epsilon$, (ii) $B = B_0 - w^{(0)}(\mathcal{S}_\epsilon)$, and (iii) the weights of facilities in $\mathcal{S}_\epsilon$ are set to zero: $w_i = 0 \;\; \forall i \in \mathcal{S}_\epsilon$ and $w_i = w_i^{(0)} \;\; \forall i \in \mathcal{F} \setminus \mathcal{S}_\epsilon$.

Lemma 4.5.1. *Suppose there is an algorithm $\mathcal{A}$ which returns a solution $\mathcal{S} = \mathcal{S}' \cup \mathcal{S}''$ for the residual instance $\mathcal{I}$ where $\mathcal{S}'' \cap \mathcal{S}_\epsilon = \emptyset$ and $c = |\mathcal{S}''|$ is a constant, $w(\mathcal{S}') \leq B$,* *and* $\mathrm{cost}_{\mathcal{I}}(\mathcal{S}) \leq \alpha \mathrm{OPT}_{\mathcal{I}}$ *for some constant* $\alpha > 0$*. Then we have that*

(i) $w^{(0)}(\mathcal{S}) \leq (1 + c\epsilon)B_0$,

(ii) $\mathrm{cost}_{\mathcal{I}_0}(\mathcal{S}) \leq \alpha \mathrm{OPT}_{\mathcal{I}_0}$.

Proof. Since all facilities in $\mathcal{S}''$ have weight at most ϵB_0, $w^{(0)}(\mathcal{S}'') \leq c\epsilon B_0$. Also, $w(\mathcal{S}') \leq B = B_0 - w^{(0)}(S_\epsilon)$ implies that $w^{(0)}(\mathcal{S}') \leq w(\mathcal{S}') + w^{(0)}(S_\epsilon) \leq B_0$. Thus,

$$w^{(0)}(\mathcal{S}) = w^{(0)}(\mathcal{S}') + w^{(0)}(\mathcal{S}'') \leq (1 + c\epsilon)B_0.$$

Now observe that $\mathrm{OPT}_{\mathcal{I}} \leq \mathrm{OPT}_{\mathcal{I}_0}$ because, by construction of $\mathcal{I}$, the chosen optimal integral of $\mathcal{I}_0$ is also a feasible solution of $\mathcal{I}$. Therefore, we have

$$\mathrm{cost}_{\mathcal{I}_0}(\mathcal{S}) = \mathrm{cost}_{\mathcal{I}}(\mathcal{S}) \leq \alpha \mathrm{OPT}_{\mathcal{I}} \leq \alpha \mathrm{OPT}_{\mathcal{I}_0}.$$

□

In the rest of this section, we aim to design such an algorithm $\mathcal{A}$ for the residual instance $\mathcal{I}$. We now compute the so-called bi-point solution as in [35].

Theorem 4.5.1. *There is a polynomial-time algorithm to compute two sets* $\mathcal{F}_1, \mathcal{F}_2 \subseteq \mathcal{F}$ *and constants* $a, b \geq 0$ *and* $a + b = 1$ *such that* $w(\mathcal{F}_1) \leq B \leq w(\mathcal{F}_2)$, $a \cdot w(\mathcal{F}_1) + b \cdot w(\mathcal{F}_2) \leq B$ *and* $a \cdot \mathrm{cost}_{\mathcal{I}}(\mathcal{F}_1) + b \cdot \mathrm{cost}_{\mathcal{I}}(\mathcal{F}_2) \leq 2\mathrm{OPT}_{\mathcal{I}}$*. (The pair* $(\mathcal{F}_1, \mathcal{F}_2)$ *is called a bi-point solution of* $\mathcal{I}$*.)*

Proof. One can extend the algorithm to construct a bi-point solution for the k-median problem in [30, 47] so that it also works for the weighted case. □

WLOG, we may assume that $d_1 \geq d_2$. Now for each client $j \in \mathcal{C}$, we let $i_1(j), i_2(j)$ denote the closest facilities to j in $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Next, we adopt the definition of *stars* as in [35]. Each facility $i_2 \in \mathcal{F}_2$ will be associated with the closest facility $i_1 \in \mathcal{F}_1$, breaking ties arbitrarily. Then, for each facility $i \in \mathcal{F}_1$, we define a *star* centered at i as a set of $\{i\}$ and its associated facilities, which will be referred to as *leaves*. Let $\mathcal{S}_i$ denote the set of leaves of the star with center i. Let $d_1(j) = d(j, i_1(j))$ and $d_2(j) = d(j, i_2(j))$. Finally, define $d_1 := \sum_{j \in \mathcal{C}} d_1(j) = \text{cost}_{\mathcal{I}}(\mathcal{F}_1)$ and $d_2 := \sum_{j \in \mathcal{C}} d_2(j) = \text{cost}_{\mathcal{I}}(\mathcal{F}_2)$.

4.5.2 Corner cases: $a \leq 1/5$ or $a \geq 4/5$

Recall that $\mathcal{F}_1$ is a feasible solution as $w(\mathcal{F}_1) \leq B$. Suppose we take $\mathcal{F}_1$ as our solution. Consider the following strategy: we connect each client j to the center i of the star $\mathcal{S}_i$ where $i_2(j) \in \mathcal{S}_i$. Fix any client j and let $\mathcal{S}_i$ be the star containing $i_2(j)$. By triangle inequality and definition of stars, we have that

$$\begin{aligned} d(j,i) &\leq d(j, i_2(j)) + d(i_2(j), i) \\ &\leq d_2(j) + d(i_2(j), i_1(j)) \\ &\leq d_2(j) + d_1(j) + d_2(j) = 2d_2(j) + d_1(j). \end{aligned}$$

Therefore, the connection cost of this assignment is bounded by $\sum_{j \in \mathcal{C}}(2d_2(j) + d_1(j)) = 2d_2 + d_1$. Now assuming we want to close the center i and open all the facilities $\mathcal{S}_i$, how does the connection cost change? Observe that the client j can now be connected to $i_2(j)$ in this case so that we can "save" the cost by $(2d_2(j) +$

$d_1(j)) - d_2(j) = d_1(j) + d_2(j)$. Our algorithm will aim to find a subset of stars such that when closing all the centers and opening all the leaves of these stars, the amount of "saving" is maximized.

For each $i \in \mathcal{F}_1$, let z_i be the indicator for the event that i is closed and all facilities in $\mathcal{S}_i$ is opened. Also, define $\delta(i) := \{j \in \mathcal{C} : i_2(j) \in \mathcal{S}_i\}$. The LP relaxation for our problem is as follows.

$$\begin{aligned} \text{LP}_{\text{star-rounding}} : \text{ maximize } & \sum_{i \in \mathcal{F}_1} \sum_{j \in \delta(i)} (d_1(j) + d_2(j)) z_i \\ \text{subject to } & \sum_{i \in \mathcal{F}_1} (w(\mathcal{S}_i) - w_i)\, z_i \le B - w(\mathcal{F}_1) \\ & 0 \le z_i \le 1 \quad \forall i \in \mathcal{F}_1. \end{aligned}$$

We are now ready to describe the algorithm. We first compute a basic optimal solution z^* of $\text{LP}_{\text{star-rounding}}$. It is not difficult to see that z^* has at most one fractional value. Suppose $z^*_{i^*}$ is fractional for some $i^* \in \mathcal{F}_1$. Consider the following LP.

$$\begin{aligned} \text{LP}_{\mathcal{S}_{i^*}} : \text{ maximize } & \sum_{i \in \mathcal{S}_{i^*}} \sum_{j \in \mathcal{C} : i_2(j) = i} (d_1(j) + d_2(j)) t_i \\ \text{subject to } & \sum_{i \in \mathcal{S}_{i^*}} w_i t_i \le w(\mathcal{S}_{i^*}) z_{i^*} \\ & 0 \le t_i \le 1 \quad \forall i \in \mathcal{S}_{i^*}. \end{aligned}$$

Next, we compute a basic optimal solution t^* for $\text{LP}_{\mathcal{S}_{i^*}}$. Again, t^* contains at most one fractional value. Suppose $t^*_{i^{**}}$ is fractional for some $i^{**} \in \mathcal{S}_{i^*}$. Let

$$\mathcal{S} := \{i^*, i^{**}\} \cup \left\{ \bigcup_{i \in \mathcal{F}_1 : z^*_i = 1} \mathcal{S}_i \right\} \cup \{i \in \mathcal{S}_{i^*} : t^*_i = 1\}.$$

Our algorithm will return the solution $\mathcal{A} := \arg\min_{Z \in \{\mathcal{S}, \mathcal{F}_1\}} \text{cost}_{\mathcal{I}}(Z)$.

Theorem 4.5.2. *If $a \leq 1/5$ or $a \geq 4/5$, then there is a bi-factor $(2.73 + \epsilon)$-approximation algorithm for* KM*.*

Proof. Let us analyze the above-mentioned algorithm. Let $\mathcal{S}'' = \{i^*, i^{**}\}$. If $w_{i^*} = 0$ or $w_{i^{**}} = 0$, then we remove the corresponding facility from $\mathcal{S}''$. Let $\mathcal{S}' = \mathcal{S} \setminus \mathcal{S}''$. By construction and the fact that z^* and t^* are feasible solutions of $\text{LP}_{\text{star-rounding}}$ and $\text{LP}_{\mathcal{S}_{i^*}}$ respectively, we have that

$$\begin{aligned} w(\mathcal{S}') &= \sum_{i \in \mathcal{F}_1 : z^*_i = 1} w(\mathcal{S}_i) + \sum_{i \in \mathcal{S}_{i^*} : t^*_i = 1} w_i \\ &\leq \sum_{i \in \mathcal{F}_1 : z^*_i = 1} w(\mathcal{S}_i) + w(\mathcal{S}_{i^*}) z^*_{i^*} \\ &\leq w(\mathcal{F}_1) + \sum_{i \in \mathcal{F}_1} (w(\mathcal{S}_i) - w_i) z^*_i \leq B. \end{aligned}$$

Next, we claim that the amount of saving by using $\mathcal{S}$ will be at least $b(d_1 + d_2)$; and hence, $\text{cost}_{\mathcal{I}}(\mathcal{S}) \leq 2d_2 + d_1 - b(d_1 + d_2) = ad_1 + (1 + a)d_2$. Observe that setting $t_i := z^*_{i^*}$ for all $i \in \mathcal{S}_{i^*}$ gives a feasible solution of $\text{LP}_{\mathcal{S}_{i^*}}$. Thus, the amount of saving by $\mathcal{S}_{i^*}$ is at least

$$\sum_{i \in \mathcal{S}_{i^*}} \sum_{j \in \mathcal{C} : i_2(j) = i} (d_1(j) + d_2(j)) z^*_{i^*} = \sum_{j \in \delta(i^*)} (d_1(j) + d_2(j)) z^*_{i^*}.$$

Then the amount of saving by $\mathcal{S}$ is at least

$$\sum_{i\in\mathcal{F}_1}\sum_{j\in\delta(i)}(d_1(j)+d_2(j))z_i^* \geq \sum_{i\in\mathcal{F}_1}\sum_{j\in\delta(i)}(d_1(j)+d_2(j))b = b(d_1+d_2),$$

because setting $z_i := b$ for all $i \in \mathcal{F}_1$ also yields a feasible solution of $\mathrm{LP}_{\text{star-rounding}}$.

Since the algorithm will choose the better solution between $\mathcal{S}$ and $\mathcal{F}_1$, we have that

$$\mathrm{cost}_{\mathcal{I}}(\mathcal{A}) \leq \min\{d_1, ad_1 + (1+a)d_2\}.$$

Then the approximation ratio with respect to the cost of the bi-point solution is

$$\begin{aligned}\min\left\{\frac{d_1}{ad_1+bd_2}, \frac{ad_1+(1+a)d_2}{ad_1+bd_2}\right\} &\leq \max_{r\geq 0, a\in[4/5,1]\vee a\in[0,1/5]} \min\left\{\frac{1}{a+(1-a)r}, \frac{a+(1+a)r}{a+(1-a)r}\right\}\\ &\leq \frac{15}{11}.\end{aligned}$$

Therefore, we have

$$\mathrm{cost}_{\mathcal{I}}(\mathcal{A}) \leq \frac{15}{11}(ad_1+bd_2) \leq \frac{30}{11}\mathrm{OPT}_{\mathcal{I}} < 2.73\mathrm{OPT}_{\mathcal{I}}.$$

Finally, the theorem follows from Lemma 4.5.1. □

4.5.3 Main case: $a \in [1/5, 4/5]$

In this section, we will apply dependent rounding in such a way that (i) each facility in $\mathcal{F}_1$ is open with probability $\approx a$, (ii) each facility in $\mathcal{F}_2$ is open with probability $\approx b$, and (iii) the budget is not violated by too much. To this end, for each $i \in \mathcal{F}_1$, let $Z_i \in \{0,1\}$ be a random indicator for the event "i is closed and all facilities in $\mathcal{S}_i$ are open." (If $Z_i = 0$ then i is open and all facilities in $\mathcal{S}_i$ are closed.) Let $t \geq 1$ be a constant to be determined. The main algorithm is as follows.

Algorithm 13 RoundStars$(t, \mathcal{F}_1, \mathcal{F}_2, a, b, B)$

```
1: Initialize $z_i \leftarrow b$ and $c_i \leftarrow w(\mathcal{S}_i) - w_i$ for all $i \in \mathcal{F}_1$
2: $\mathbf{Z} \leftarrow$ SRDR$(\mathbf{z}, \mathbf{c}, t)$
3: Let $\mathcal{F}' := \{i \in \mathcal{F}_1 : 0 < Z_i < 1\}$ be the set of $\leq t$ left-over fractional values of $\mathbf{Z}$
4: Let $\mathcal{L} := \bigcup_{i \in \mathcal{F}'} \mathcal{S}_i$
5: for each $i \in \mathcal{L}$: set $r_i \leftarrow Z_k$ where $k \in \mathcal{F}'$ is the center of the star containing $i$
6: $\mathbf{R} \leftarrow$ SRDR$(\mathbf{r}, \mathbf{w}, t)$
7: Let $\mathcal{L}' := \{i \in \mathcal{L} : 0 < R_i < 1\}$ be the set of $\leq t$ left-over fractional values of $\mathbf{R}$
8: return $\mathcal{S} = \mathcal{F}' \cup \mathcal{L}' \cup \left\{\bigcup_{i \in \mathcal{F}_1 : Z_i = 1} \mathcal{S}_i\right\} \cup \{i \in \mathcal{L} : R_i = 1\}$
```

We will now analyze the algorithm RoundStars.

Claim 4.5.1. *We have that $|\mathcal{F}' \cup \mathcal{L}'| \leq 2t$ and $w(\mathcal{S} \setminus (\mathcal{F}' \cup \mathcal{L}')) \leq B$.*

Proof. The first claim is trivial due to construction of $\mathcal{F}'$ and $\mathcal{L}'$. By the property of SRDR, we have

$$\sum_{i \in \mathcal{L} : R_i = 1} w_i \leq \sum_{i \in \mathcal{L}} w_i R_i = \sum_{k \in \mathcal{F}'} w(\mathcal{S}_k) Z_k.$$

Therefore,

$$\begin{aligned}
w(\mathcal{S} \setminus (\mathcal{F}' \cup \mathcal{L}')) &= \sum_{i \in \mathcal{F}_1 : Z_i = 1} w(\mathcal{S}_i) + \sum_{i \in \mathcal{L} : R_i = 1} w_i \\
&\leq \sum_{i \in \mathcal{F}_1 : Z_i = 1} w(\mathcal{S}_i) + \sum_{k \in \mathcal{F}'} w(\mathcal{S}_k) Z_k \\
&= \sum_{i \in \mathcal{F}_1} w(\mathcal{S}_i) Z_i \\
&\leq w(\mathcal{F}_1) + \sum_{i \in \mathcal{F}_1} (w(\mathcal{S}_i) - w_i) Z_i \\
&= w(\mathcal{F}_1) + \sum_{i \in \mathcal{F}_1} (w(\mathcal{S}_i) - w_i) b \\
&= a w(\mathcal{F}_1) + b w(\mathcal{F}_2) \leq B,
\end{aligned}$$

where the penultimate equality follows from the fact that SRDR preserves the weighted sum of Z_i's. □

Now let us fix a client j and analyze the expected connection cost of j. Note that RoundStars guarantees that whenever a center i of some star is closed, all of the leaves in $\mathcal{S}_i$ will be open. Thus, we will use the following strategy to get j connected: if $i_2(j)$ is open, then assign j to $i_2(j)$. Else, if $i_1(j)$ is open, assign j to $i_1(j)$. Finally, if both $i_1(j)$ and $i_2(j)$ are closed, the center of the star containing $i_2(j)$, say i, must be open and we connect j to i. For simplicity, let i_1, i_2 denote the event that $i_1(j)$ and $i_2(j)$ are open respectively.

Claim 4.5.2. *The following bounds hold:*

- $\Pr[\bar{i}_2] \leq 1 - b$,

- $\Pr[\bar{i}_1\bar{i}_2] \leq (1+2/t)b(1-b)$.

Proof. Conditioned on any vector Z returned by SRDR, if $Z_i \in \{0,1\}$ then $\Pr[\bar{i}_2] = 1 - Z_i$. If $i \in \mathcal{F}'$, conditioned on any vector R returned by SRDR, we have $\Pr[i_2] \geq R_{i_2(j)}$ because $i_2(j)$ will be open even if $R_{i_2(j)}$ is fractional. This implies that $\Pr[i_2] \geq \mathbf{E}[R_{i_2(j)}] = Z_i$ or $\Pr[\bar{i}_2] \leq 1 - Z_i$. Summing over all possible Z's, we get

$$\Pr[\bar{i}_2] \leq \mathbf{E}[1 - Z_i] = 1 - z_i = 1 - b.$$

Again, conditioned on any vector Z, we claim that

$$\Pr[\bar{i}_1\bar{i}_2] \leq Z_{i_1(j)}(1 - Z_i).$$

Indeed, if $Z_{i_1(j)} = 0$ or $Z_{i_1(j)} \in (0,1)$, then $i_1(j)$ will be opened by the algorithm and $\Pr[\bar{i}_1\bar{i}_2] = 0$. If $Z_{i_1(j)} = 1$, then $i_1(j)$ will be closed and $\Pr[\bar{i}_1\bar{i}_2] = \Pr[\bar{i}_2] = 1 - Z_i = Z_{i_1(j)}(1 - Z_i)$ as in the above case. Thus, summing over all possible Z's, we obtain

$$\begin{aligned}
\Pr[\bar{i}_1\bar{i}_2] &\leq \mathbf{E}[Z_{i_1(j)}(1 - Z_i)] \\
&\leq (z_{i_1(j)}(1 - z_i))^{1-1/(t+1)} \\
&= (b(1-b))^{1-1/(t+1)},
\end{aligned}$$

where the second inequality is due to the near independence property of SRDR. Since $a \in [1/5, 4/5]$, we have $b \in [1/5, 4/5]$ and $b(1-b) \geq 4/25$. Hence, $(b(1 -$

$b))^{-1/(t+1)} \leq (4/25)^{-1/(t+1)} \leq 1 + 2/t$. Therefore, we get

$$\Pr[\bar{i}_1\bar{i}_2] \leq (1 + 2/t)b(1 - b).$$

$\square$

Now the expected connection cost j can be bounded as follows.

$$\begin{aligned}
\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(j)] &\leq \Pr[i_2]d_2(j) + \Pr[i_1\bar{i}_2]d_1(j) + \Pr[\bar{i}_1\bar{i}_2](2d_2(j) + d_1(j)) \\
&= \Pr[i_2]d_2(j) + \Pr[\bar{i}_2]d_1(j) + 2\Pr[\bar{i}_1\bar{i}_2]d_2(j) \\
&= d_2(j) + (d_1(j) - d_2(j))\Pr[\bar{i}_2] + 2\Pr[\bar{i}_1\bar{i}_2]d_2(j) \\
&\leq d_2(j) + (d_1(j) - d_2(j))(1 - b) + (1 + 2/t)b(1 - b)(2d_2(j)).
\end{aligned}$$

Summing over all clients j, we get

$$\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{S})] \leq d_2 + (d_1 - d_2)(1 - b) + (1 + 2/t)b(1 - b)(2d_2).$$

Theorem 4.5.3. *If $a \in [1/5, 4/5]$, then there is a bi-factor $(1+\sqrt{3}+\epsilon)$-approximation algorithm for* KM.

Proof. First of all, we remove all facilities i for which $w_i = 0$ from $\mathcal{F}'$ and $\mathcal{L}'$. Observe that Claim 4.5.1 still holds. It means that we can write $\mathcal{S} = \mathcal{S}' \cup \mathcal{S}''$ where $\mathcal{S}'' = \mathcal{F}' \cup \mathcal{L}'$, $\mathcal{S}' = \mathcal{S} \setminus \mathcal{S}'$, $|\mathcal{S}''| \leq 2t$, $w(\mathcal{S}') \leq B$ and $\mathcal{S}''$ does not contain "big facilities": $\mathcal{S}'' \cup \mathcal{S}_\epsilon = \emptyset$. Then, by Lemma 4.5.1, we have

$$w^{(0)}(\mathcal{S}) \leq (1 + 2t\epsilon)B_0.$$

Now suppose we will take the better solution between $\mathcal{S}$ and $\mathcal{F}_1$ (which is feasible). In other words, let us consider the set $\mathcal{A} := \arg\min_{Z \in \{\mathcal{S}, \mathcal{F}_1\}} \mathrm{cost}_{\mathcal{I}}(Z)$. We have

$$\begin{aligned}
\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{A})] &\leq \min\{d_1, d_2 + (d_1 - d_2)(1-b) + (1 + 2/t)b(1-b)(2d_2)\} \\
&\leq (1 + 2/t)\min\{d_1, d_2 + (d_1 - d_2)(1-b) + b(1-b)(2d_2)\} \\
&= (1 + 2/t)(ad_1 + bd_2)\min\left\{\frac{d_1}{ad_1 + bd_2}, \frac{d_2 + (d_1 - d_2)(1-b) + b(1-b)(2d_2)}{ad_1 + bd_2}\right\} \\
&\leq (1 + 2/t) \cdot (2\mathrm{OPT}_{\mathcal{I}}) \cdot C_0,
\end{aligned}$$

where

$$\begin{aligned}
C_0 &= \min\left\{\frac{d_1}{ad_1 + bd_2}, \frac{d_2 + (d_1 - d_2)(1-b) + b(1-b)(2d_2)}{ad_1 + bd_2}\right\} \\
&= \min\left\{\frac{d_1}{(1-b)d_1 + bd_2}, \frac{d_2 + (d_1 - d_2)(1-b) + b(1-b)(2d_2)}{(1-b)d_1 + bd_2}\right\} \\
&\leq \max_{r \in [0,1],\ b \in [1/5, 4/5]} \min\left\{\frac{1}{1 - b + br}, \frac{r + (1-r)(1-b) + b(1-b)(2r)}{1 - b + br}\right\} \\
&\leq \frac{1 + \sqrt{3}}{2}.
\end{aligned}$$

Therefore, we get

$$\begin{aligned}\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{A})] &\le (1+2/t)\cdot(1+\sqrt{3})\cdot \mathrm{OPT}_{\mathcal{I}} \\ &\le (1+6/t)\cdot \mathrm{OPT}_{\mathcal{I}}.\end{aligned}$$

In conclusion, for any $\gamma > 0$, setting $t = 6/\gamma$ and $\epsilon = \gamma/(2t) = \gamma^2/12$ gives

(i) $w^{(0)}(\mathcal{A}) \le (1+\gamma)B_0$,

(ii) $\mathbf{E}[\mathrm{cost}_{\mathcal{I}}(\mathcal{A})] \le (1+\gamma)\mathrm{OPT}_{\mathcal{I}_0}$.

$\square$

Chapter 5: The (Multi) Knapsack Center Problem

5.1 Problem definition

In this chapter, we consider the Multi Knapsack Center problem (MKC). An instance $\mathcal{I}$ of this problem consists of a set V of n vertices, a symmetric distance metric d on V, and an $m \times n$ weight matrix M, which corresponds to m non-negative weight functions. We assume that all the weights are scaled so that the corresponding budgets are equal to one and the entries of M are in $[0, 1]$. Our goal is to choose a set $\mathcal{S} \subseteq V$ of vertices (which will also be called "centers") so that (a) all m knapsack constraints are satisfied — that is, we require that

$$\sum_{i \in S} M_{ki} \leq 1,$$

for all $k = 1, \ldots, m$ and (b) the maximum connection cost of any vertex (equivalently, the radius for centers in $\mathcal{S}$ to cover all vertices in V)

$$R := \max_{j \in V} \operatorname{cost}(j) = \max_{j \in V} \min_{i \in \mathcal{S}} d(i, j)$$

is minimized.

5.2 Prior work and Our contributions

The Knapsack Center (KC) problem (i.e., the case $m = 1$ knapsack constraint) was first studied by Hochbaum and Shmoys in [54], under the name "weighted k-center problem". The authors gave a 3-approximation algorithm for the problem and proved that this is best possible unless P = NP; see also [58]. More recently, Chen et. al. [59] considered the above general version with m knapsack constraints. They showed that this problem is not approximable to within any constant factor, and gave a pseudo 3-approximation algorithm which may violate all but one knapsack constraint by a factor of $(1 + \epsilon)$.

Given any instance $\mathcal{I}$ of the MKC problem, we let R be the optimal radius. For the standard KC problem with one constraint, we give a polynomial-time algorithm which returns a feasible solution such that (1) all vertices are within distance $3R$ from some chosen center and (2) almost all vertices have expected connection cost at most $(1 + 2/e)R \approx 1.74R$. We refer this as the *fair* knapsack-center algorithm.

For the MKC problem, we show that it is possible to obtain a similar result while slightly violating the knapsack constraints via independent rounding. (Again, the violation is likely unavoidable because it is NP-hard to approximate this problem to within any constant factor.)

5.3 Preliminaries

Note that there are only $\binom{n}{2}$ possible values for the optimal radius R. Thus, we can *guess* this value in $O(n^2)$ time. For the rest of this chapter, we assume that R is the *correct* optimal radius. Consider the polytope $\mathcal{P}(\mathcal{I}, R)$ containing points (x, y), where $x \in \mathbb{R}^{n\times n}$ and $y \in \mathbb{R}^n$, with the following constraints:

(A1) $\sum_{i\in V:d(i,j)\leq R} x_{ij} = 1$ for all $j \in V$, (all clients should get connected to some center)

(A2) $x_{ij} \leq y_i$ for all $i, j \in V$, (vertex j can only connect to center i if it is open)

(A3) $My \leq \vec{1}$, (the m knapsack constraints)

(A4) $0 \leq x_{ij}, y_i \leq 1$ for all $i, j \in V$.

Note that y_i and x_{ij} are indicators for the event whether center i is opened and whether j is connected to i, respectively. By a (standard) trick of splitting the facilities in Facility-Location problems (see, e.g., [28,50,53]), we may further assume that

(A5) For all $i, j \in V$, we have $x_{ij} \in \{0, y_i\}$,

(A6) For all $i \in V$, we have $x_{ii} = y_i$.

We provide a proof of this claim here for completeness.

Claim 5.3.1. *Given any instance $\mathcal{I}$ of the* MKC *problem and a fractional solution $(x, y) \in \mathcal{P}(\mathcal{I}, R)$, one can construct another instance $\mathcal{I}'$ with the set of vertices V' of size $O(n^2)$ and another solution (x', y') such that* *(i)* $(x', y') \in \mathcal{P}(\mathcal{I}', R)$,

(ii) Any solution of $\mathcal{I}'$ *can be converted into a solution of* $\mathcal{I}$ *without increasing the objective function,*

(iii) For all $i, j \in V'$*, we have* $x'_{ij} \in \{0, y'_i\}$,

(iv) For all $i \in V'$*, we have* $x'_{ii} = y'_i$.

Proof. We will construct $\mathcal{I}'$ and (x', y') as follows. Initially, let $V' := V$. For each $i \in V$, let $j_1, j_2, \ldots, j_n \in V$ be such that $x_{ij_1} \leq x_{ij_2} \leq \ldots \leq x_{ij_n}$. Now for each j_k in this order, we add a vertex i_k into V' (if $j_k = i$ then simply let $i_k := i$ and note that $i_k \in V'$), which is co-located at i and has the same weight as i in each of the m knapsack constraints. We then set $y'_{i_k} := x_{ij_k} - x_{ij_{k-1}}$ if $k > 1$ and $y'_{i_k} := x_{ij_k}$ otherwise. Also, we set $x'_{i_{k'}j_k} := y'_{i_{k'}}$ for $k' \in [1, k]$ and $x'_{i_{k'}j_k} := 0$ for $k' \in [k+1, n]$. (Note that $j_k \in V$ which implies that $j_k \in V'$. Thus, these assignments are valid.) Let M' denote the new $m \times n^2$ weight matrix of vertices in V'.

In the above step, for each of the original vertex $i \in V$, we have added $n - 1$ additional copies of i into V' and already defined the corresponding assignment variables x' for it. Now, for each copy i_k of i, we simply connect i_k to the same vertices serving i fractionally: $x'_{vi_k} := x'_{vi}$ for all $v \in V'$. By construction, the properties (iii) and (iv) are satisfied.

Observe that each $i \in V$ has n copies in V'. For any solution of $\mathcal{I}'$, if we simply remove the extra $\leq n - 1$ copies if they are open, then we obtain a feasible solution of $\mathcal{I}$ with the same cost. Thus, property (ii) is satisfied.

We now verify that $(x', y') \in \mathcal{P}(\mathcal{I}', R)$. Again, by construction, we have $x'_{ij}, y'_i \in [0, 1]$ and $x'_{ij} \leq y'_i$ for all $i, j \in V'$. Fix any $j \in V' \cap V$. For any $i \in V$, suppose x_{ij} is the t_{ij}-th smallest value when we process vertex i in the above step. Note that

$$\begin{aligned}\sum_{k=1}^{t_{ij}} y'_{i_k} &= x_{ij_1} + (x_{ij_2} - x_{ij_1}) + \ldots + (x_{ij_{t_{ij}}} - x_{ij_{t_{ij}-1}}) \\ &= x_{ij_{t_{ij}}} = x_{ij}.\end{aligned}$$

Then we have

$$\begin{aligned}\sum_{i \in V' : d(i,j) \leq R} x'_{ij} &= \sum_{i \in V : d(i,j) \leq R} \sum_{k=1}^{n} x'_{i_k j} \\ &= \sum_{i \in V : d(i,j) \leq R} \sum_{k=1}^{t_{ij}} x'_{i_k j} \\ &= \sum_{i \in V : d(i,j) \leq R} \sum_{k=1}^{t_{ij}} y'_{i_k} \\ &= \sum_{i \in V : d(i,j) \leq R} x_{ij} = 1.\end{aligned}$$

By construction, this equality also holds for all other copies of j in V'.

Now fix vertex $i \in V$. Recall that all copies of i in V' have the same weight as i in each of the knapsack constraint. Fix any knapsack constraint and suppose w is the weight of i in this constraint. The contribution of all copies of i in this constraint is

$$\begin{aligned} w\sum_{k=1}^{n} y'_{i_k} &= w(x_{ij_1} + (x_{ij_2} - x_{ij_1}) + \ldots + (x_{ij_n} - x_{ij_{n-1}})) \\ &= wx_{ij_n} \le wy_i. \end{aligned}$$

Therefore, we have $M'y' \le My \le \vec{1}$.

$\square$

For the rest of this chapter, we assume that properties (A5) and (A6) hold.

Given any LP solution satisfying (A1) – (A6), our goal is to (randomly) round y to an integral solution. For any $j \in V$, let $F_j := \{i \in V : x_{ij} > 0\}$. We refer to these sets as *clusters*. It is easy to verify that $y(F_j) = 1$ due to (A5). We now form a subset $V' \subseteq V$ such that all the clusters $F_j, F_{j'}$ for $j, j' \in V'$ are pairwise disjoint, and such that V' is maximal with this property. Let $F_0 := \{i \in V \mid y_i > 0, j \notin \bigcup_{j \in V'} F_j\}$. We note that, by Property (A6), we have that $j \in F_j$ if $y_j > 0$, which implies that F_0 and V' are disjoint.

Then we partition V into a set of *groups*. There are two types of groups. First, for each $j \in V$, we define the group G_j to be simply F_j. Next, for each $j \in F_0$, we create a group G_j which consists of two items, namely j and a new dummy item Dummy(j), which has $y(\text{Dummy}(j)) = 1 - y_j$, distance ∞ from all $i \in V$, and zero weights in all m knapsack constraints. (If we select this dummy item, it simply means that we do not choose to select item $j \in F_0$.)

The following notation will be useful throughout: for any set $X \subseteq [n]$, we define frac(X) to be the set of items $i \in X$ such that $y_i \notin \{0,1\}$; here y will always denote the *current* value of the vector $y \in [0,1]^n$.

In the first step of our algorithms, we simplify y to reduce the number of fractional items in each G_j to at most $m+1$ (this is automatically the case for $j \in F_0$). This can be done by the following procedure KnapsackIntragroupReduce.

Algorithm 14 KnapsackIntragroupReduce (y, X)

```
1: while $|\text{frac}(X)| > m+1$ do
2:     Let $\delta \in \mathbb{R}^n, \delta \neq 0$ be such that $M\delta = 0$, $\delta(X) = 0$, and $\delta_i = 0 \ \ \forall i \notin \text{frac}(X)$
3:     Choose scaling factors $a, b > 0$ such that
         • $y + a\delta \in [0,1]^n$ and $y - b\delta \in [0,1]^n$
         • there is at least one entry of $y + a\delta$ which is equal to zero or one
         • there is at least one entry of $y - b\delta$ which is equal to zero or one
4:     With probability $\frac{b}{a+b}$, update $y \leftarrow y + a\delta$; else, update $y \leftarrow y - b\delta$.
5: return $y$
```

Claim 5.3.2. *One can find a vector $\delta \in \mathbb{R}^n$ as claimed in line 2 of* KnapsackIntragroupReduce.

Proof. This line is only executed when we have at least $m+2$ variables in X. On the other hand, we have only m knapsack constraints ($M\delta = 0$) and the additional linear constraint $\delta(X) = 0$. This system is underdetermined and the claim follows. □

Claim 5.3.3. *Suppose $y' =$ KnapsackIntragroupReduce(y, X) for any $X \subseteq [n]$. Then we have $y'(X) = y(X)$ and $My = My'$ and $E[y'_i] = y_i$ for all $i \in V$.*

Proof. In each round of KnapsackIntragroupReduce, we update y by either $y' := y + a\delta$ or $y' := y - b\delta$. Since δ is chosen so that $\delta(X) = 0$, we have $y'(X) = y(X) + a\delta(X) = y(X)$ or $y'(X) = y(X) - b\delta(X) = y(X)$. Similarly, as $M\delta = 0$ we have $My = My'$.

Also, for any $i \in V$ we have

$$\mathbf{E}[y_i'] = y_i + \frac{b}{a+b}(a\delta_i) - \frac{a}{a+b}(b\delta_i) = y_i.$$

The claim follows by induction on all iterations. □

5.4 A *fair* knapsack-center algorithm for the case $m = 1$

Now we show that, when $m = 1$ (the standard Knapsack Center problem), one can satisfy the knapsack constraint with no violation while guaranteeing that the expected approximation ratio of at least $(1-\delta)n$ vertices is at most $1 + 2/e + \gamma$ for any $\gamma > 0$.

Theorem 5.4.1. *For any $\delta, \gamma > 0$, there exists an algorithm running in $n^{O\left(\frac{1}{\delta\gamma^2}\right)}$ time and returns a feasible solution the Knapsack Center problem (i.e., with $m = 1$ knapsack constraint) such that* $\text{cost}(j) \leq 3R$ *for all $j \in V$. Moreover, there is a set $U \subseteq V$ (which is deterministic, not a random variable), such that:*

1. *$|U| \geq (1-\delta)n$; and*
2. *$\forall j \in U$, we have* $\mathbf{E}[\text{cost}(j)] \leq (1 + 2/e + \gamma)R$.

5.4.1 Algorithm

High-level ideas: First, by a preprocessing step, we find a fractional solution in which each vertex i with $y_i > 0$ will only serve (fractionally) at most ϵn other vertices. Next, we use the procedure KNAPSACKINTRAGROUPREDUCE to reduce the number of fractional variables inside G_j down to 2 for all $j \in W$. Note that the opening mass $y(G_j) = 1$ remains unchanged in the process. Define X_j to be the indicator for choosing the fractional vertex of higher weight in G_j. We then use SRDR to round vector $\mathbf{X}$ into an "almost" integral solution. That is, there will be at most some $O(1)$ groups G_j containing exactly two fractional vertices. Finally, we open the vertex with *smaller* weight in each fractional group.

We let M denote the (only) weight function in this problem. In this subsection, slightly abusing the notation, we shall also think of M as a vector where M_i denotes the weight of node $i \in V$.

Algorithm 15 PRUNE($\epsilon, M, V, \mathcal{S}, R$)

```
 1: if $V = \emptyset$ then
 2:    return $\mathcal{S}$    // we obtain an optimal solution in this case
 3: $\mathcal{I}' \leftarrow (M, V)$
 4: if $\mathcal{P}(\mathcal{I}', R) \neq \emptyset$ then
 5:    Compute a solution $(x, y) \in \mathcal{P}(\mathcal{I}', R)$
 6:    if there exists some center $i$ such that $M_i \leq 1$ and $|\{j \in V : x_{ij} > 0\}| \geq \epsilon n$
       then
 7:       Let $X \leftarrow \{j \in V : x_{ij} > 0\}$
 8:       return PRUNE($\epsilon, \frac{M}{1-M_i}, V \setminus X, \mathcal{S} \cup \{i\}, R$) if it is not FALSE
 9:       return PRUNE($\epsilon, M, V \setminus \{i\}, \mathcal{S}, R$))
10:    else
11:       return (the solution $(x, y)$, the set of remaining vertices $V$, the scaled
          weight matrix $M$, and the set of open centers $\mathcal{S}$)
12: else
13:    return FALSE
```

The main algorithm is as follows.

(We note that it is formally possible for this process to return $\mathcal{S}$ which contains some dummy items. As these dummy items have infinite distance and zero weight, they contribute nothing and can simply be discarded.)

Algorithm 16 StandardKnapsackSRDR (M, V, t)

```
1: $(x, y, V, M, \mathcal{S}_0) \leftarrow$ Prune$(\epsilon, M, V, \emptyset, R)$ and **return** $\mathcal{S}$ if it already covers all nodes in $V$
2: **for each** $j \in V'$ **do** update $y \leftarrow$KnapsackIntragroupReduce$(y, G_j)$;
3: **for each** $j \in W$ that $|\text{frac}(G_j)| = 2$ **do**
4:     Suppose $G_j = \{i_1(j), i_2(j)\}$ and $M_{i_1(j)} \leq M_{i_2(j)}$
5:     Set $x_j \leftarrow y_{i_2(j)}$ and $a_j \leftarrow M_{i_2(j)} - M_{i_1(j)}$
6: $\mathbf{X} \leftarrow \text{SRDR}(\mathbf{x}, \mathbf{a}, t)$
7: **for each** $j \in W$ that $|\text{frac}(G_j)| = 2$ **do**
8:     Set $y_{i_1(j)} \leftarrow 1 - X_j$ and $y_{i_2(j)} \leftarrow X_j$ // *for the sake of simpler analysis*
9: **for each** $j \in W$ that $|\text{frac}(G_j)| = 2$ **do**
10:    Set $y_{i_1(j)} \leftarrow 1$ and $y_{i_2(j)} \leftarrow 0$
11: **return** $\mathcal{S} = \{i \in V : y_i = 1\} \cup \mathcal{S}_0$
```

5.4.2 Analysis

Lemma 5.4.1. *The procedure* Prune *runs in $n^{O(1/\epsilon)}$ time and will return either an optimal solution or a set $\mathcal{S}$ of open centers along with a fractional solution (x, y) for the residual instance $\mathcal{I}' = (V, d, m)$, in which each center i only serves $\leq \epsilon n$ other vertices fractionally. That is, $|\{j \in V : x_{ij} > 0\}| \leq \epsilon n$ for all $i \in V$.*

Proof. Fix any optimal solution to the given instance. In Algorithm 15, whenever we find a vertex i which serves $\geq \epsilon n$ other vertices, we simply guess two cases: whether or not i is in the optimal solution. Note that for the guess "i is in the optimal solution", we open i and remove at least ϵn other vertices from the instance. (Because we assume that the budget constraint has RHS value of 1, this requires rescaling M to $\frac{M}{1-M_i}$.)

In the other case, we remove i from V. Observe that if our guess is correct, $\mathcal{P}(\mathcal{I}', R)$ will not be empty in the next step. The algorithm stops when the current fractional solution satisfies all the properties in the claim.

To bound the running time of this algorithm, we can visualize its execution by a binary tree in which each node is either a vertex chosen in line 6 or a leaf. Each non-leaf node of the tree has two children corresponding to two decisions whether or not it is in the optimal solution. Indeed, the running time of the algorithm is bounded by the number of paths from the root to any leaf of this tree. Because (i) the length of any path is at most n and (ii) the number of vertices chosen to be in the optimal solution in this path is at most $1/\epsilon$, the number of such paths is at most $n^{O(1/\epsilon)}$.

□

For the rest of this subsection, we will be working on the residual instance (M, V) returned by Prune. In the first part of the analysis, we show an upper bound on the probability that a given vertex $k \in V$ has no open facility in F_k. This is done by defining a natural potential function, which is an estimation for this probability. We will analyze its change after applying SRDR.

For each $j \in W$ we let $C_j = F_k \cap G_j$ be the set of vertices that vertex k is "interested in." Let y denote the current fractional solution at the beginning of line 3 of StandardKnapsackSRDR. We define a potential function:

$$S := \prod_{j \in W} \left(1 - y(C_j)\right).$$

Let y' be the modified vector y after finishing the for-loop at lines 7–8 and

$$S' := \prod_{j \in W} \left(1 - y'(C_j)\right).$$

Lemma 5.4.2. *Conditioned on any value of y and S, we have that*

$$\mathbf{E}[S'] \leq S^{1-1/(t+1)}.$$

Proof. Recall that the algorithm KnapsackIntragroupReduce will reduce the number of fractional values in all G_j's to at most 2. Also, $y(G_j) = y'(G_j) = 1$ for all $j \in W$ by Claim 5.3.3. The claim follows immediately from the following observations:

- If there exists $j \in W$ such that $y(C_j) = 1$, then $S = S' = 0$ and the equality happens. So let us assume that such a vertex does not exist.
- Any client $j \in W$ for which $y(C_j) = 0$ will have no contribution to both S and S'.
- Now let us focus on clients $j \in W$ for which $y(C_j) \in (0,1)$. Since $|G_j| = 2$ and $y(G_j) = 1$, we have that either $i_1(j) \in C_j$ or $i_2(j) \in C_j$ but not both. Let $A = \{j \in W : i_1(j) \in C_j\}$ and $B = \{j \in W : i_2(j) \in C_j\}$. Observe that

$A \cap B = \emptyset$ and

$$\begin{aligned}
S &= \prod_{j \in W} (1 - y(C_j)) \\
&= \prod_{j \in A} (1 - y(C_j)) \prod_{j \in B} (1 - y(C_j)) \\
&= \prod_{j \in A} (1 - y_{i_1(j)}) \prod_{j \in B} (1 - y_{i_2(j)}) \\
&= \prod_{j \in A} y_{i_2(j)} \prod_{j \in B} y_{i_1(j)}.
\end{aligned}$$

Then we have

$$\begin{aligned}
\mathbf{E}[S'] &= \mathbf{E}\left[\prod_{j \in A} (1 - y'(C_j)) \prod_{j \in B} (1 - y'(C_j))\right] \\
&= \mathbf{E}\left[\prod_{j \in A} (1 - y_{i_1(j)}) \prod_{j \in B} (1 - y_{i_2(j)})\right] \\
&= \mathbf{E}\left[\prod_{j \in A} X_j \prod_{j \in B} (1 - X_j)\right] \\
&\leq \mathbf{E}\left[\left(\prod_{j \in A} X_j \prod_{j \in B} (1 - X_j)\right)^p\right] \\
&\leq \left(\prod_{j \in A} x_j \prod_{j \in B} (1 - x_j)\right)^p \\
&= \left(\prod_{j \in A} y_{i_2(j)} \prod_{j \in B} y_{i_1(j)}\right)^p = S^p,
\end{aligned}$$

where $p = 1 - 1/(t + 1)$ and the second inequality follows from the near-independence property of SRDR.

□

Lemma 5.4.3. *We have that*

$$\mathbf{E}[S'] \leq (1/e)^{1-1/(t+1)}.$$

Proof. Let y^0 be the original fractional solution y at line 1 and $S_0 := \prod_{j\in W}(1 - y_0(C_j))$. We have $S_0 \leq \prod_{j\in W} e^{-\sum_{i\in C_j} y_i^0} = e^{-y^0(F_k)} = 1/e$. Next, since the clusters G_j are processed independently in line 1, and marginals are preserved by Claim 5.3.3, we have

$$\mathbf{E}[S] = \prod_{j\in W}(1 - \mathbf{E}[y_i^1]) = \prod_{j\in W}(1 - y_i^0) = S_0 \leq 1/e.$$

By Lemma 5.4.2 and Jensen's inequality, we have

$$\mathbf{E}[S'] \leq \mathbf{E}[S^{1-1/(t+1)}] \leq \mathbf{E}[S]^{1-1/(t+1)} \leq (1/e)^{1-1/(t+1)}.$$

□

Proof of Theorem 5.4.1. **Feasibility of $\mathcal{S}$:** Recall that after calling the procedure Prune at line 1, the fractional solution (x, y) is feasible for the residual instance (V, M). It suffices to show that we do not violate the scaled knapsack constraint of the residual instance when rounding y. Claim 5.3.3 ensures that the knapsack constraint is preserved (fractionally) by KnapsackIntraGroupReduce. Suppose that y and y' are the fractional solution at the beginning of line 3 and line 9, respectively. Let $J = \{j \in W : |\text{frac}(G_j)| = 2\}$ denote the set of nodes considered in the for-loop at line 3. We have

$$\begin{aligned}\sum_{j\in J}(M_{i_1(j)}y_{i_1(j)} + M_{i_2(j)}y_{i_2(j)}) &= \sum_{j\in J}(M_{i_1(j)}(1-x_j) + M_{i_2(j)}x_j)\\ &= \sum_{j\in J}(M_{i_1(j)} + a_j x_j)\\ &= \sum_{j\in J}(M_{i_1(j)} + a_j X_j)\\ &= \sum_{j\in J}(M_{i_1(j)}y'_{i_1(j)} + M_{i_2(j)}y'_{i_2(j)}),\end{aligned}$$

where the third equality follows from the sum preservation property of SRDR. This means that the contribution of clients $j \in J$ to the knapsack constraint remains the same at the beginning of line 9. Now, when rounding the last t fractional clusters G_j in the for-loop at lines 9–10, we always open the node $i_1(j)$ and close $i_2(j)$ where $M_{i_1(j)} \leq M_{i_2(j)}$. Since $y_{i_1(j)} + y_{i_2(j)} = 1$, we have that $M_{i_1(j)}$ is bounded by the fractional contribution of both $i_1(j)$ and $i_2(j)$ to the knapsack constraint.

Cost analysis: For any $k \in V$, we have $\text{cost}(k) \leq R$ if there is a facility opened in F_k, and $\text{cost}(k) \leq 3R$ otherwise. (By construction, $F_k \cap F_j \neq \emptyset$ for some $j \in V'$ and we always open one center inside G_j. Then the distance from k to this center is at most $d(k,j) + R \leq 3R$ by triangle inequality.) Note that there are at most $2t$ fractional vertices in V after applying SRDR. For each vertex j, let q_j

denote the probability that j is still adjacent to such an unrounded vertex. Then

$$\begin{aligned}
\sum_{j\in V} q_j &\le \sum_{j\in V}\sum_{k\in F_j} \Pr[\text{vertex } k \text{ is fractional}] \qquad \text{(by the union-bound)}\\
&= \sum_{k\in V} \Pr[\text{vertex } k \text{ is fractional}] \sum_{j\in F_k} 1\\
&\le (\epsilon n)\sum_{k\in V} \Pr[\text{vertex } k \text{ is fractional}] \qquad \text{(by the pre-processing step)}\\
&\le (2t)\epsilon n.
\end{aligned}$$

We say that a vertex j is good if $q_j \le 1/t$ and bad otherwise. Then the number of bad vertices is at most $2t^2\epsilon n$. We let U be the set of good vertices. Let y'' be the vector y after finishing the for-loop at lines 9–10 and let

$$S'' := \prod_{j\in W} (1 - y''(C_j)).$$

Now fix any $k \in U$. We have that $F_k \cap \mathcal{S} = \emptyset$ iff $S'' = 1$, and $S'' = 0$ otherwise. Let $\mathcal{E}$ denote the event that there are no fractional vertices in F_k before line 9. Note that $\Pr[\mathcal{E}] = 1 - q_k \geq 1 - 1/t$. Then, for t large enough, we have

$$\begin{aligned}
\Pr[\mathcal{S} \cap F_k = \emptyset] &= \Pr[\mathcal{S} \cap F_k = \emptyset \wedge \mathcal{E}] + \Pr[\mathcal{S} \cap F_k = \emptyset \wedge \neg\mathcal{E}] \\
&\leq \Pr[S'' = 1 \wedge \mathcal{E}] + \Pr[\neg\mathcal{E}] \\
&\leq \Pr[S' = 1 \wedge \mathcal{E}] + 1/t \qquad (\text{if } \mathcal{E} \text{ then } S' = S'') \\
&\leq \mathbf{E}[S'] + 1/t \\
&\leq (1/e)^{1-1/(t+1)} + 1/t \\
&\leq 1/e + 2/t,
\end{aligned}$$

where we use the fact that $e^{1/(t+1)} \leq 1 + 2/(t+1)$ in the last inequality. Thus, we get

$$\mathbf{E}[\text{cost}(k)] \leq R + (2R)\Pr[\mathcal{S} \cap F_k = \emptyset] \leq (1 + 2/e + 2/t)R.$$

Now for any $\delta, \gamma > 0$, by setting $t = 2/\gamma$ and $\epsilon = \frac{\delta}{2t^2}$, the number of bad vertices is $\leq \delta n$ and, for any $k \in U$, we have $\mathbf{E}[\text{cost}(k)] \leq (1 + 2/e + \gamma)R$. The running time is $n^{O(1/\epsilon)} = n^{O\left(\frac{1}{\delta\gamma^2}\right)}$.

□

5.5 Independent rounding algorithm for the case $m \geq 1$

In this section, we will introduce a simple *fair* algorithm for the MKC problem based on independent rounding. The idea is to first get rid of *big* vertices which have large weight in some constraint. Then all the weights in the residual instance will be relative *small*. This allows us to randomly pick a center in each cluster F_j

using the distribution defined by the opening variables. By Chernoff bound, we will slightly violate some knapsack constraint with high probability.

To this end, we will first enforce the following additional constraint on the maximum size of the entries of M.

Claim 5.5.1. *For any $\rho > 0$, we can find a solution $(x, y) \in \mathcal{P}(\mathcal{I}, R)$ with properties (A1) to (A6), which also satisfies the additional property*

(A7) For any $i \in V$, if $M_{ki} \geq \rho$ for any $k = 1, \ldots, m$, we have $y_i \in \{0, 1\}$.

Also, the running time for this process in $n^{O(m/\rho)}$.

Proof. We say that the vertex i is *big* if $M_{ki} \geq \rho$ for some $k = 1, \ldots, m$. Suppose we fix any optimal solution $\mathcal{S}$. Observe that there can be at most m/ρ big centers in $\mathcal{S}$. We can guess the set of such centers in time $n^{O(m/\rho)}$. For any big center i that we guess is in $\mathcal{S}$, we set $y_i := 1$. Similarly, for any big center that we guess is outside $\mathcal{S}$, we set $y_i := 0$. This procedure will only check at most $n^{O(m/\rho)}$ possible cases. If our guess is correct, we have that $\mathcal{P}(\mathcal{I}, R) \neq \emptyset$ with the additional constraints on y. $\square$

Suppose (x, y) satisfies all properties (A1) to (A7). The *fair* algorithm for the MKC problem is as follows.

Algorithm 17 INDEPENDENTROUND (y, V', M)

```
1: $\mathcal{S} \leftarrow \emptyset$
2: for $i \in F_0$ do
3:     With probability $y_i$, $\mathcal{S} \leftarrow \mathcal{S} \cup \{i\}$
4: for $j \in V'$ do
5:     Randomly pick a vertex $X_j$ in $F_j$ s.t. $\Pr[X_j = i] = y_i$ for all $i \in F_j$ // recall that $y(F_j) = 1$
6:     $\mathcal{S} \leftarrow \mathcal{S} \cup \{X_j\}$
7: return $\mathcal{S}$
```

For any vertex $i \in V$, let Y_i be the indicator variable for the event that $i \in \mathcal{S}$. By construction, we have $\mathbf{E}[Y_i] = y_i$ and all variables Y_i are negatively correlated.

Claim 5.5.2. *For any $\epsilon \in (0,1)$, with probability at least $1 - m \exp\left(-\frac{\epsilon^2}{3\rho}\right)$, we have that $\sum_{i \in \mathcal{S}} M_{ki} \leq 1 + \epsilon$ for all $1 \leq k \leq m$.*

Proof. The k^{th} weight function $\sum_{i \in \mathcal{S}} M_{ki} = M_k Y$ is a sum of negatively-correlated variables, each of which is bounded in $[0, \rho]$ and which has mean at most 1. Since the Chernoff-Hoeffding bound holds under negative correlation, we have

$$\Pr[M_k Y \geq 1 + \epsilon] = \Pr\left[\frac{M_k Y}{\rho} \geq (1+\epsilon) \times \frac{1}{\rho}\right] \leq e^{-\epsilon^2/(3\rho)}.$$

Taking a union bound over all m constraints, the total probability that any of them is violated by more than ϵ is at most $m e^{-\epsilon^2/(3\rho)}$. □

Lemma 5.5.1. *For any vertex $j \in V$, we have the conditional expectation*

$$\mathbf{E}[\text{cost}(j) \mid MY \leq \vec{1} + \epsilon] \leq \left(1 + \frac{2/e}{1 - m\exp(-\epsilon^2/(3\rho))}\right) R.$$

Proof. Fix any vertex $j \in V$. Note that $y(F_j) = 1$. So by negative correlation, the probability that there are no open centers in F_j is

$$\begin{aligned}\Pr[F_j \cap \mathcal{S} = \emptyset] &= \mathbf{E}\left[\prod_{i \in F_j} (1 - Y_i)\right] \\ &\leq \prod_{i \in F_j} (1 - y_i) \leq e^{-y(F_j)} = 1/e.\end{aligned}$$

Then, by Claim 5.5.2, we have

$$\begin{aligned}\Pr[F_j \cap \mathcal{S} = \emptyset \mid MY \leq \vec{1} + \epsilon] &= \frac{\Pr[F_j \cap \mathcal{S} = \emptyset \wedge MY \leq \vec{1} + \epsilon]}{\Pr[MY \leq \vec{1} + \epsilon]} \\ &\leq \frac{\Pr[F_j \cap \mathcal{S} = \emptyset]}{\Pr[MY \leq \vec{1} + \epsilon]} \\ &\leq \frac{1/e}{1 - m \exp\left(-\frac{\epsilon^2}{3\rho}\right)}.\end{aligned}$$

Recall that $\text{cost}(j) \leq 3R$ with probability one. Therefore,

$$\begin{aligned}\mathbf{E}[\text{cost}(j) \mid MY \leq \vec{1} + \epsilon] \leq R + \frac{1}{e\,(1 - m \exp(-\epsilon^2/(3\rho)))} \times (2R) \\ = \left(1 + \frac{2/e}{1 - m \exp(-\epsilon^2/(3\rho))}\right) R.\end{aligned}$$

$\square$

Theorem 5.5.1. *For any $0 < \gamma \leq 1/2$ and $0 < \epsilon < 1$, there is an algorithm running in expected time $n^{O(m \log(m/\gamma)\epsilon^{-2})}$, which outputs a solution $\mathcal{S}$ satisfying*

1. $MY \leq \vec{1} + \epsilon$*,*

2. $\forall j \in V: \ \mathbf{E}[\text{cost}(j)] \leq (1 + 2/e + O(\gamma))R.$

Proof. Set $\rho = \epsilon^2/(3 \log(m/\gamma))$ and apply Claim 5.5.1 to achieve a fractional solution y satisfying (A1) – (A7). Now repeatedly assign $\mathcal{S} := \textsc{IndependentRound}(y, V', M)$ until we obtain a solution $\mathcal{S}$ satisfying $MY \leq \vec{1} + \epsilon$.

Observe that the resulting distribution on the random variables $\text{cost}(j)$ output by this process is the same as the distribution of $\text{cost}(j)$, conditioned on $MY \leq \vec{1}+\epsilon$.

Applying Lemma 5.5.1, for any $j \in V$, we get

$$\mathbf{E}[\mathrm{cost}(j)] \leq \left(1 + \frac{2/e}{1 - m\exp(-\epsilon^2/(3\rho))}\right) R.$$

By our choice of ρ, we have $m \exp\left(-\frac{\epsilon^2}{3\rho}\right) \leq \gamma$ and hence

$$\mathbf{E}[\mathrm{cost}(j)] \leq \left(1 + \frac{2/e}{1-\gamma}\right) R \leq (1 + 2/e + O(\gamma))R.$$

Also, the number of repetitions of this process is a geometric random variable, with success probability equal to the probability that $MY \leq \vec{1} + \epsilon$, which is at least $1 - \gamma \geq 1/2$. So we only need an expected constant number of iterations to succeed. Therefore, the overall running time is $n^{O(m/\rho)} = n^{O(m \log(m/\gamma)\epsilon^{-2})}$. □

Chapter 6: The k-center Problem

6.1 Problem definition

In this chapter, we will discuss a *fair* algorithm for the k-center problem. Recall that the k-center problem is a special case of the MKC problem where we only have one cardinality constraint instead of m knapsack constraints. In particular, an instance $\mathcal{I} = (V, d, k)$ of this problem consists of set V of n vertices, a symmetric distance metric d on V, and a parameter $k \in \mathbb{N}$.

Our objective is to choose a set $\mathcal{S} \subseteq V$ of vertices (which will also be called "centers") so that (a) at most k centers are opened: $|\mathcal{S}| \leq k$ and (b) the maximum connection cost of any vertex (equivalently, the radius for centers in $\mathcal{S}$ to cover all vertices in V)

$$R := \max_{j \in V} \operatorname{cost}(j) = \max_{j \in V} \min_{i \in \mathcal{S}} d(i, j)$$

is minimized. We shall refer to centers in $\mathcal{S}$ as open centers.

6.2 Prior work and Our contributions

The k-center problem is known to be NP-hard via a reduction from the Dominating Set problem (see, e.g., [89]). The problem can be approximated to within a factor of 2 by very simple greedy algorithms [54, 90, 91]. On the other hand, this is also the best possible approximation ratio one can obtain unless P=NP [92].

To the best of our knowledge, all the current approximation algorithms for this problem in the literature are deterministic. For any such algorithm $\mathcal{A}$, it is not difficult to point out an instance for which the connection cost of almost all vertices assigned by $\mathcal{A}$ matches the worst case bound (i.e., 2 times the optimal radius.)

Our result here will be a *fair* algorithm. Suppose we are given an instance $\mathcal{I} = (V, d, k)$ of the problem, and suppose we have guessed the optimal radius R. We show that there is a randomized polynomial-time algorithm that opens at most k centers with probability one while guaranteeing that (1) all vertices are within distance $3R$ from some chosen center and (2) the expected distance $\mathbf{E}[\text{cost}(j)]$ from any given vertex j to the nearest open center is at most $1.597R$.

We leave it as an open question whether one can improve the worst-case guarantee from $3R$ to $2R$ while still achieving the expected ratio less than 2.

6.3 Preliminaries

Recall that there are only $\binom{n}{2}$ possible values for the optimal radius R and we can guess this value R in $O(n^2)$ time. Consider the polytope $\mathcal{P}(\mathcal{I}, R)$ containing points (x, y), where $x \in \mathbb{R}^{n \times n}$ and $y \in \mathbb{R}^n$, with the following constraints:

(A1) $\sum_{i \in V : d(i,j) \leq R} x_{ij} = 1$ for all $j \in V$, (all vertices should get connected to some center)

(A2) $x_{ij} \leq y_i$ for all $i, j \in V$, (vertex j can only connect to center i if it is open)

(A3) $\sum_{i \in V} y_i \leq k$, (at most k centers are opened)

(A4) $0 \leq x_{ij}, y_i \leq 1$ for all $i, j \in V$.

Since R is the optimal radius, $\mathcal{P}(\mathcal{I}, R)$ is not empty. Our approach will be to find a fractional solution in $\mathcal{P}(\mathcal{I}, R)$ and then use a randomized algorithm to convert it into an integral solution.

By splitting vertices as needed, we can ensure that we have a fractional solution which satisfies the additional properties

(A5) For all $i, j \in V$, we have $x_{ij} \in \{0, y_i\}$,

(A6) For all $i \in V$, we have $x_{ii} = y_i$.

For any $j \in V$, let $F_j := \{j\} \cup \{i \in V : x_{ij} > 0\}$. We refer to these sets as *clusters*, and we refer to j as the *cluster center* of the cluster F_j. By (A5), (A6), and (A1), we have $y(F_j) = 1$ for all $j \in V$.

6.4 A simple algorithm with expected ratio of 1.6

6.4.1 Algorithm

In this section, we will give a simple randomized rounding scheme based on forming clusters centered around certain vertices.

In the first scheme, we let $V' \subseteq V$ be a set of vertices which has the property that all F_j for $j \in V'$ are pairwise disjoint, and such that V' is maximal with this property. (This can be formed easily in a greedy way.) We define $F_0 := V \setminus \bigcup_{j \in V'} F_j$. This is called the set of "unclustered" vertices.

One natural idea is to open a random center inside each cluster F_j (all clusters are processed independently) and apply dependent rounding to choose $y(F_0)$ centers in F_0. Now for each vertex j, one can show that the bad event where no center in F_j is chosen is at most $1/e$. In such an event, one can still connect j to some open center in F_k where $F_k \cap F_j \neq \emptyset$ at distance $\leq 3R$ from j. Thus, we have $\mathbf{E}[\text{cost}(j)] \leq (1 - 1/e)R + (1/e)3R = (1 + 2/e)R \approx 1.73R$.

To improve the expected ratio from 1.73 to 1.6, we employ the following idea. Let $q \in [0, 1]$ be a parameter to be determined. For each cluster F_j where $j \in V'$, we will open the cluster center j with probability q. With the remaining probability $1-q$, we randomly open a center in F_j using the distribution defined by the opening variables y_i. Intuitively, this will increase the chance that any vertex j can connect to some cluster center k at distance $\leq 2R$ from k. The formal algorithm is as follows.

Algorithm 18 Round1 $\left(y, F_0, \bigcup_{j \in V'} F_j, q\right)$

```
1: $\mathcal{S} \leftarrow \emptyset$
2: for $j \in V'$ do
3:     Randomly pick a vertex $X_j \in F_j$ and assign $\mathcal{S} \leftarrow \mathcal{S} \cup \{X_j\}$ according to the
       following distribution

       $$\forall i \in F_j : \Pr[X_j = i] = \begin{cases} q + (1-q)y_i & \text{if } i = j \\ (1-q)y_i & \text{if } i \neq j \end{cases}$$

       // This is a valid probability distribution, as $\sum_{i \in F_j} y_i = y(F_j) = 1$
4: Let $I_0 \leftarrow$ DepRound$(y, F_0)$
5: $\mathcal{S} \leftarrow \mathcal{S} \cup I_0$
6: return $\mathcal{S}$
```

6.4.2 Analysis

Throughout this chapter, we let Y_i be an indicator variable for the event that center i is open.

Claim 6.4.1. Round1 *returns a solution $\mathcal{S}$ of at most k centers.*

Proof. Since the clusters F_j are pairwise disjoint and $y(F_j) = 1$, we have

$$\sum_{j \in V'} y(F_j) = |V'|.$$

Observe that the dependent rounding procedure ensures that

$$|I_0| \leq \lceil y(F_0) \rceil = \lceil \sum_{j \in V} y_i - \sum_{j \in V'} y(F_j) \rceil \leq k - |V'|.$$

Therefore, we have $|\mathcal{S}| = |V'| + |\mathcal{I}_0| \leq |V'| + k - |V'| = k$. □

Theorem 6.4.1. *For $q = 0.464587$,* Round1 *returns a solution $\mathcal{S}$ such that, for any $j \in V$, we have*

1. $\text{cost}(j) \leq 3R$,
2. $\mathbf{E}[\text{cost}(j)] \leq 1.60793R$.

Proof. Note that, for any $j \in V'$, there will be some open center F_j; and hence, $\text{cost}(j) \leq R$. Fix any vertex $j \in V \setminus V'$. Let D denote the set of all $i \in V'$ such that $F_i \cap F_j \neq \emptyset$. ($D$ is the set cluster centers which are "close" to j.) By maximality of V', we must have $F_i \cap F_j \neq \emptyset$ for some $i \in V'$, which implies $D \neq \emptyset$.

For each $i \in D$, let $m_i := y(F_i \cap F_j)$ and let $m_0 := y(F_j \cap F_0)$. As F_0 and F_i's (where $i \in V'$) are all pairwise disjoint, we have $m_0 + \sum_{i \in D} m_i = y(F_i) = 1$.

For each $i \in D$, our rounding step opens exactly one center $v \in F_i$. As every center in F_j has distance at most R to j, and all centers in F_i has distance at most $2R$ from each other, it follows that $d(j, v) \le 3R$. Note that

$$F_j = (F_0 \cap F_j) \cup \bigcup_{i \in D} (F_i \cap F_j).$$

By negative correlation, we have

$$\begin{aligned}
\Pr[\mathrm{cost}(j) \ge 2R] &\le \Pr[\text{no centers in } F_j \text{ are open}] \\
&= \mathbf{E}\left[\prod_{i \in D}\left(1 - \sum_{v \in F_i \cap F_j} Y_v\right) \prod_{v \in F_j \cap F_0} (1 - Y_v)\right] \\
&\le \prod_{i \in D}\left(1 - \sum_{v \in F_i \cap F_j} \mathbf{E}[Y_v]\right) \prod_{v \in F_j \cap F_0} (1 - \mathbf{E}[Y_v]) \\
&\le \prod_{i \in D}\left(1 - \sum_{v \in F_i \cap F_j} (1-q) y_v\right) \prod_{v \in F_j \cap F_0} (1 - y_v) \\
&\le \prod_{i \in D} (1 - (1-q) m_i) \prod_{v \in F_j \cap F_0} e^{-y_v} \\
&= \prod_{i \in D} (1 - (1-q) m_i) \times e^{-1 + \sum_{i \in D} m_i} \\
&= (1/e) \prod_{i \in D} e^{m_i} (1 - (1-q) m_i),
\end{aligned}$$

where in the second inequality, we use the fact that $\mathbf{E}[Y_v] \ge (1-q) y_v$.

Similarly, if for some $i \in D$ we open center i itself, then $d(i,j) \leq 2R$ and hence $\mathrm{cost}(j) \leq 2R$. A necessary condition for $\mathrm{cost}(j) \geq 3R$ is that we do not open any center in $F_j \cup D = (F_0 \cap F_j) \cup \bigcup_{i \in D}(F_i \cap F_j) \cup D$.

$$\begin{aligned}
\Pr[\mathrm{cost}(j) \geq 3R] &\leq \Pr[\text{no centers in } F_j \cup D \text{ are open }] \\
&= \mathbf{E}\left[\prod_{i \in D}\left(1 - Y_i - \sum_{v \in F_i \cap F_j \setminus \{i\}} Y_v\right) \prod_{v \in F_j \cap F_0}(1 - Y_v)\right] \\
&\leq \prod_{i \in D}\left(1 - q - \sum_{v \in F_i \cap F_j \setminus \{i\}}(1-q)y_v\right) \prod_{v \in F_j \cap F_0}(1 - y_v) \\
&\leq \prod_{i \in D}(1 - q - (1-q)m_i) \prod_{v \in F_j \cap F_0} e^{-y_v} \\
&= (1/e)\prod_{i \in D} e^{m_i}(1 - q - (1-q)m_i).
\end{aligned}$$

Thus, we have that

$$\begin{aligned}
\mathbf{E}[\mathrm{cost}(j)] &\leq R(\Pr[\mathrm{cost}(j) \geq R] + \Pr[\mathrm{cost}(j) \geq 2R] + \Pr[\mathrm{cost}(j) \geq 3R]) \\
&\leq R\left(1 + (1/e)\prod_{i \in D} e^{m_i}(1 - (1-q)m_i) + (1/e)\prod_{i \in D} e^{m_i}(1 - q - (1-q)m_i)\right).
\end{aligned}$$

Let $m := \sum_{i \in D} m_i$, $t = |D|$, and $p = 1 - q = 0.53542$. Then, by AM-GM inequality, we have

$$\begin{aligned}
\mathbf{E}[\mathrm{cost}(j)] &\leq R\left(1 + e^{m-1}\prod_{i \in D}(1 - pm_i) + e^{m-1}\prod_{i \in D}(p - pm_i)\right) \\
&\leq R\left(1 + e^{m-1}(1 - pm/t)^t + e^{m-1}(p - pm/t)^t\right).
\end{aligned}$$

Let $f(m,t) := e^{m-1}(1-pm/t)^t + e^{m-1}(p-pm/t)^t$. Recall that $m \in [0,1]$ and $t \in \mathbb{N}$. For $t \in \{1,2,3,4\}$, it is not difficult to check the $f(m,t)$ achieves the maximum value ≈ 0.60792 at $(m = 0.43386, t = 1)$ and $(m = 1, t = 2)$. For $t \geq 5$, using the fact that $(1 + x/n) \leq e^x$ for any $n > 1$ and $|x| \leq n$, we have

$$\begin{aligned}
f(m,t) &= e^{m-1}(1-pm/t)^t + e^{m-1}(p-pm/t)^t \\
&\leq e^{m-1-pm} + e^{m-1} \cdot p^t e^{-m} \\
&= e^{1-0.53542m} + p^t/e \\
&\leq e^{1-0.53542m} + 0.0161875 \\
&\leq 0.601611.
\end{aligned}$$

Therefore, we conclude that $\mathbf{E}[\text{cost}(j)] \leq 1.60792R$. □

6.5 Improved algorithm using partial clusters

6.5.1 Algorithm

Here we discuss an improvement on the previous algorithm by using *partial clusters*. Observe that so far we have only used *full clusters*: each cluster F_j for $j \in V'$ has opening mass exactly one (i.e., $y(F_j) = 1$.) For each such cluster, we have moved some mass of the whole cluster towards increasing the chance of opening its center. Therefore, for any vertex $j \in V \setminus V'$, the chance that j gets connected to some open center at distance $\leq 2R$ is improved. One of the worst-case scenarios in the previous analysis is when F_j intersects with only a few full clusters. In this case, if we can move some mass from $F_j \cap F_0$ towards opening the center j itself, the expected connection cost of j will be improved. Roughly speaking, the set $F_j \cap F_0$ will be a *partial cluster* with center j.

To this end, we use the following greedy algorithm to form both full and partial clusters.

Algorithm 19 GreedyFormClusters (y)

```
1: Set $U \leftarrow V$ and $F'_j \leftarrow F_j$ for all $j \in V$
2: Set $\mathcal{G} \leftarrow \emptyset$ and $\ell \leftarrow 0$
3: **while** $y(U) > 0$ **do**
4:     $\ell \leftarrow \ell + 1$
5:     Find $j \in U$ such that $y(F'_j)$ is maximized
6:     Set $G_\ell \leftarrow F'_j, \mathcal{G} \leftarrow \mathcal{G} \cup G_\ell$, $z_\ell \leftarrow y(F'_j)$, and $\pi(\ell) \leftarrow j$
7:     Set $U \leftarrow U \setminus G_\ell$
8:     **for each** $i \in U$ **do** $F'_i \leftarrow F'_i \setminus G_\ell$
9: **return** $\mathcal{G}$, vector $z$, and vector $\pi$
```

We define $g := |\mathcal{G}|$ be the number of clusters. Now, for $\ell \in [1, g]$, we shall refer to G_ℓ as a *full cluster* if $z_\ell = 1$. Otherwise, G_ℓ is called a *partial cluster*. In both cases, $\pi(\ell)$ is called the cluster center of G_ℓ. Let p_{full} and $p_{\text{partial}} \in [0, 1]$ be two random parameters to be determined. The main algorithm is as follows.

Algorithm 20 ROUND2 $(y, p_{\text{full}}, p_{\text{partial}})$

```
1: Run GREEDYFORMCLUSTERS($y$) to obtain $G_\ell$'s and $z$
2: $\mathcal{S} \leftarrow \emptyset$
3: $Z \leftarrow$ DEPROUND($z$)
4: for $\ell \in Z$ do
5:     Randomly pick a vertex $X_\ell \in G_\ell$ and assign $\mathcal{S} \leftarrow \mathcal{S} \cup \{X_\ell\}$ according to the
       following distribution
           $$\forall i \in G_j : \Pr[X_\ell = i] = \begin{cases} q_\ell + (1-q_\ell)y_i/z_\ell & \text{if } i = \pi(\ell) \\ (1-q_\ell)y_i/z_\ell & \text{if } i \neq \pi(\ell) \end{cases}$$
       and where we define $q_\ell$ as
           $$q_\ell = \begin{cases} p_{\text{full}} & \text{if } z_\ell = 1 \\ p_{\text{partial}} & \text{if } z_\ell < 1 \end{cases}$$
6: return $\mathcal{S}$
```

6.5.2 Analysis

Note that the distribution defined at line 5 of ROUND2 is valid. It is clear that $\Pr[X_\ell = i] \geq 0$ and

$$\sum_{i \in G_\ell} \Pr[X_\ell = i] = q_\ell + \sum_{i \in G_\ell} (1-q_\ell)y_i/z_\ell$$
$$= q_\ell + (1-q_\ell)(1/z_\ell)y(G_\ell) = 1.$$

Claim 6.5.1. *The clusters in $\mathcal{G}$ returned by Algorithm* GREEDYFORMCLUSTERS *are pairwise disjoint and $y(V) = \sum_{\ell=1}^{g} y(G_\ell)$. Moreover, we have*

1. *$z_1 = 1$,*

2. *for any $1 \leq \ell < g$, we have $z_\ell \geq z_{\ell+1}$,*

Proof. We maintain the invariant that U is set of the current uncovered vertices.

By construction, all the cluster G_ℓ's are pairwise disjoint and cover all vertices in U having positive opening mass. The stopping condition of the while-loop at line 3 is $y(U) = 0$, which implies that $y(V) = \sum_{\ell=1}^{g} y(G_\ell)$. Since $y(F_j) = 1$ for any $j \in V$, we have that $z_1 = 1$. The second property holds by our greedy choice of j. □

We now claim that the algorithm always outputs a feasible solution.

Claim 6.5.2. *Algorithm* Round2 *returns a solution $\mathcal{S}$ satisfying $|\mathcal{S}| \le k$.*

Proof. By construction, we have

$$\sum_{\ell=1}^{g} z_\ell = \sum_{\ell=1}^{g} y(G_\ell) = y(G_v) = \sum_{i \in V} y_i \le k.$$

Since dependent rounding preserves, we get $|Z| \le \lceil \sum_{\ell=1}^{n} z_\ell \rceil \le k$. The claim follows because we only exactly one center for each cluster G_ℓ where $\ell \in Z$ in the while loop at lines 4–5.

□

Lemma 6.5.1. *For any $j \in V$, we have that* $\text{cost}(j) \le 3R$ *with probability one.*

Proof. If j is the cluster center of a full cluster then we have indeed $\text{cost}(j) \le R$. Otherwise, there must exist a full cluster G_ℓ such that $G_\ell \cap F_j \neq \emptyset$. (Else, F_j itself should have been added into $\mathcal{G}$ by the algorithm as a full cluster.) Let i be the open center in G_ℓ. By triangle inequality, $d(i, j) \le 3R$. □

Claim 6.5.3. *For any $T \subseteq V$, we have*

$$\Pr[\mathcal{S} \cap T = \emptyset \mid p_{\text{full}}, p_{\text{partial}}] \le \prod_{\ell=1}^{g} \left(1 - (1 - q_\ell) y(T \cap G_\ell) - q_\ell z_\ell [\pi(\ell) \in T]\right).$$

Proof. In this proof, let us condition on a fixed pair of $(p_{\text{full}}, p_{\text{partial}})$. The event $\mathcal{S} \cap T = \emptyset$ means that there are no open centers in T. Now consider cluster G_ℓ's. Suppose Z_ℓ is the event that $\ell \in Z$ (i.e., $\Pr[Z_\ell] = z_\ell$.) Let $\mathcal{E}_\ell$ denote the event that $\mathcal{S} \cap (G_\ell \cap T) \neq \emptyset$. Observe that Z_ℓ's are negatively correlated. Moreover, $\Pr[\mathcal{E}_\ell | Z_\ell] \geq 0$, $\Pr[\mathcal{E}_\ell | \bar{Z}_\ell] = 0$, and all $\mathcal{E}_\ell$'s are independent conditioned on any Z. Thus, the event $\mathcal{E}_\ell$'s are also negatively correlated. We have

$$\begin{aligned}
\Pr[\mathcal{S} \cap T = \emptyset] &= \Pr\left[\bigwedge_{\ell=1}^{g} [\mathcal{S} \cap (G_\ell \cap T) = \emptyset]\right] \\
&= \Pr\left[\bigwedge_{\ell=1}^{g} \bar{\mathcal{E}}_\ell\right] \\
&\leq \prod_{\ell=1}^{g} \Pr\left[\bar{\mathcal{E}}_\ell\right] \\
&= \prod_{\ell=1}^{g} (\Pr[\ell \notin Z] + \Pr[\ell \in Z](1 - (1 - q_\ell) y(G_\ell \cap T)/z_\ell - q_\ell[\pi(\ell) \in T])) \\
&= \prod_{\ell=1}^{g} (1 - z_\ell + z_\ell(1 - (1 - q_\ell) y(G_\ell \cap T)/z_\ell - q_\ell[\pi(\ell) \in T])) \\
&= \prod_{\ell=1}^{g} (1 - (1 - q_\ell) y(G_\ell \cap T) - q_\ell z_\ell[\pi(\ell) \in T]).
\end{aligned}$$

where the first equality is due to the fact that only vertices in T having positive opening mass can be opened and such vertices must belong to some cluster.

□

Now let us consider some worst-case instance $\mathcal{I}$ and the vertex $j \in V$ for which $\mathbf{E}[\text{cost}(j)]$ is as large as possible. We can assume WLOG that (i) $y_j = 0$ and (ii) $j \in U$ when the algorithm terminates (i.e., all vertices in F_j are claimed by other clusters in GreedyFormClusters.) Indeed, if this is not the case, we can simply add another vertex j' collocated at j with $y_{j'} = 0$ and $x_{ij'} = x_{ij}$ for all i. Then j' satisfies the desired properties.

Let $J_{\mathrm{f}}, J_{\mathrm{p}} \subseteq [n]$ be the sets of clusters which intersect with F_j:

$$J_{\mathrm{f}} := \{\ell \in [g] : F_j \cap G_\ell \neq \emptyset, z_\ell = 1\},$$

$$J_{\mathrm{p}} := \{\ell \in [g] : F_j \cap G_\ell \neq \emptyset, z_\ell < 1\}.$$

Suppose that $|J_{\mathrm{f}}| = r$ and suppose that J_{p} is sorted as $J_{\mathrm{p}} = \{\ell_1, \ldots, \ell_t\}$ where $\ell_1 < \ell_2 < \cdots < \ell_t$. For each $s = 1, \ldots, t+1$, we define the "suffix" sum:

$$u_s = y(F_j \cap G_{\ell_s}) + y(F_j \cap G_{\ell_{s+1}}) + \ldots + y(F_j \cap G_{\ell_t}).$$

Then we have $1 \geq u_1 \geq u_2 \geq \cdots \geq u_t \geq u_{t+1} = 0$, $r \geq 1$.

Claim 6.5.4. *For all $s = 1, \ldots, t$, we have $z_{\ell_s} \geq u_s$.*

Proof. Recall that $z_{\ell_s} = y(G_{\ell_s})$ is the maximum mass we can have from the remaining clusters at time ℓ_s. On the other hand, at time ℓ_s, the vertex j is still in U and $y(F'_j) \geq y(F_j \cap G_{\ell_s}) + y(F_j \cap G_{\ell_{s+1}}) + \ldots + y(F_j \cap G_{\ell_t}) = u_s$. Thus, $z_{\ell_s} \geq y(F'_j) \geq u_s$. □

Lemma 6.5.2. *We have that*

$$\mathbf{E}[\mathrm{cost}(j) \mid p_{\mathrm{full}}, p_{\mathrm{partial}}] \leq R\Bigg(1 + \left(1 - \frac{(1-p_{\mathrm{full}})(1-u_1)}{r}\right)^r \prod_{s=1}^{t}(1-(1-p_{\mathrm{partial}})(u_s - u_{s+1})) + \left(1 - p_{\mathrm{full}} - \frac{(1-p_{\mathrm{full}})(1-u_1)}{r}\right)^r \prod_{s=1}^{t}(1-u_s+(1-p_{\mathrm{partial}})u_{s+1})\Bigg).$$

Proof. Again let us condition on a fixed pair of $(p_{\mathrm{full}}, p_{\mathrm{partial}})$. in this proof. For any ℓ, let $m_\ell := y(F_j \cap G_\ell)$. Note that $\sum_{\ell \in J_{\mathrm{f}} \cup J_{\mathrm{p}}} m_\ell = y(F_j) = 1$. Hence, $\sum_{\ell \in J_{\mathrm{f}}} m_\ell = 1 - \sum_{\ell \in J_{\mathrm{p}}} m_\ell = 1 - u_1$. By Claim 6.5.3, we have

$$\begin{aligned}
\Pr[\mathrm{cost}(j) \geq 2R] &\leq \Pr[\mathcal{S} \cap F_j = \emptyset] \\
&\leq \prod_{\ell=1}^{g}(1-(1-q_\ell)m_\ell - q_\ell z_\ell[\pi(\ell) \in T]) \\
&\leq \prod_{\ell=1}^{g}(1-(1-q_\ell)m_\ell) \\
&= \prod_{\ell \in J_{\mathrm{f}}}(1-(1-p_{\mathrm{full}})m_\ell) \cdot \prod_{s=1}^{t}(1-(1-p_{\mathrm{partial}})m_{\ell_s}) \\
&= \prod_{\ell \in J_{\mathrm{f}}}(1-(1-p_{\mathrm{full}})m_\ell) \cdot \prod_{s=1}^{t}(1-(1-p_{\mathrm{partial}})(u_s - u_{s+1})) \\
&\leq \left(1 - \frac{(1-p_{\mathrm{full}})\sum_{\ell \in J_{\mathrm{f}}} m_\ell}{r}\right)^r \cdot \prod_{s=1}^{t}(1-(1-p_{\mathrm{partial}})(u_s - u_{s+1})) \\
&= \left(1 - \frac{(1-p_{\mathrm{full}})(1-u_1)}{r}\right)^r \cdot \prod_{s=1}^{t}(1-(1-p_{\mathrm{partial}})(u_s - u_{s+1})),
\end{aligned}$$

where we use the AM-GM inequality in the penultimate inequality.

Similarly, a necessary condition for $\mathrm{cost}(j) \geq 3R$ is that $\mathcal{S} \cap T = \emptyset$ where

$T := F_j \cup \{\pi(\ell) : G_\ell \cap F_j \neq \emptyset\}$. By Claim 6.5.3, we get

$$\begin{aligned}
\Pr[\mathrm{cost}(j) \geq 3R] &\leq \Pr[\mathcal{S} \cap T = \emptyset] \\
&\leq \prod_{\ell=1}^{g} (1 - (1-q_\ell)m_\ell - q_\ell z_\ell[\pi(\ell) \in T]) \\
&= \prod_{\ell \in J_{\mathrm{f}}} (1 - (1-p_{\mathrm{full}})m_\ell - p_{\mathrm{full}} z_\ell) \cdot \prod_{s=1}^{t} (1 - (1-p_{\mathrm{partial}})m_{\ell_s} - p_{\mathrm{partial}} z_{\ell_s}) \\
&= \prod_{\ell \in J_{\mathrm{f}}} (1 - (1-p_{\mathrm{full}})m_\ell - p_{\mathrm{full}}) \cdot \prod_{s=1}^{t} (1 - (1-p_{\mathrm{partial}})(u_s - u_{s+1}) - p_{\mathrm{partial}} z_{\ell_s}) \\
&\leq \left(1 - p_{\mathrm{full}} - \frac{(1-p_{\mathrm{full}})\sum_{\ell \in J_{\mathrm{f}}} m_\ell}{r}\right)^r \cdot \prod_{s=1}^{t} (1 - (1-p_{\mathrm{partial}})(u_s - u_{s+1}) \\
&\qquad - p_{\mathrm{partial}} u_s) \\
&= \left(1 - p_{\mathrm{full}} - \frac{(1-p_{\mathrm{full}})(1-u_1)}{r}\right)^r \cdot \prod_{s=1}^{t} (1 - u_s + (1-p_{\mathrm{partial}})u_{s+1}),
\end{aligned}$$

where we use the AM-GM inequality and the fact that $z_{\ell_s} \geq u_s$ in the penultimate inequality. The claim follows as $\mathbf{E}[\mathrm{cost}(j)] \leq R(\Pr[\mathrm{cost}(j) \geq R] + \Pr[\mathrm{cost}(j) \geq 2R] + \Pr[\mathrm{cost}(j) \geq 3R])$. $\square$

By Lemma 6.5.2, we have

$$\mathbf{E}[\mathrm{cost}(j)] \leq R\mathbf{E}_p[(1 + AB + CD)],$$

where the expectation $\mathbf{E}_p$ is taken over random choices of $(p_{\text{full}}, p_{\text{partial}})$, and

$$\begin{aligned}
A &:= \left(1 - \frac{(1-p_{\text{full}})(1-u_1)}{r}\right)^r, \\
B &:= \prod_{s=1}^{t}(1-(1-p_{\text{partial}})(u_s - u_{s+1})), \\
C &:= \left(1 - p_{\text{full}} - \frac{(1-p_{\text{full}})(1-u_1)}{r}\right)^r, \\
D &:= \prod_{s=1}^{t}(1-u_s+(1-p_{\text{partial}})u_{s+1}).
\end{aligned}$$

Note that $\mathbf{E}_p[(1+AB+CD)]$ is the expected approximation ratio of our algorithm and we now want to find a good upper-bound for it. The main difficulty is that $(AB+CD)$ is a function of r, t and $1 \geq u_1 \geq u_2 \ldots \geq u_t \geq 0$, and r, t can be arbitrarily large. We approximate this by using only the first t_0 elements of the sequence u and truncating r by a small, fixed value r_0. To this end, let us fix two small constants $r_0, t_0 > 0$. We will now upper-bound A, B, C, and D.

If $r < r_0$, we use the current formulas for A and C. If $r \geq r_0$, we will upper-bound A and C as

$$\begin{aligned}
A &= \left(1 - \frac{(1-p_{\text{full}})(1-u_1)}{r}\right)^r \\
&\leq \exp(-(1-p_{\text{full}})(1-u_1)),
\end{aligned}$$

and

$$\begin{aligned}
C &= \left(1 - p_{\text{full}} - \frac{(1-p_{\text{full}})(1-u_1)}{r}\right)^r \\
&= (1-p_{\text{full}})^r \left(1 - \frac{1-u_1}{r}\right)^r \\
&\leq (1-p_{\text{full}})^{r_0} \exp(u_1 - 1).
\end{aligned}$$

Suppose we define $u_{t+1} = u_{t+2} = \ldots = 0$. The bounds for B and D are as follows.

$$\begin{aligned}
B &= \prod_{s=1}^{\infty}(1-(1-p_{\text{partial}})(u_s - u_{s+1})) \\
&= \prod_{s=1}^{t_0-1}(1-(1-p_{\text{partial}})(u_s - u_{s+1})) \cdot \prod_{s=t_0}^{\infty}(1-(1-p_{\text{partial}})(u_s - u_{s+1})) \\
&\leq \prod_{s=1}^{t_0-1}(1-(1-p_{\text{partial}})(u_s - u_{s+1})) \cdot \prod_{s=t_0}^{\infty}\exp(-(1-p_{\text{partial}})(u_s - u_{s+1})) \\
&= \prod_{s=1}^{t_0-1}(1-(1-p_{\text{partial}})(u_s - u_{s+1})) \cdot \exp\left(-(1-p_{\text{partial}})\sum_{s=t_0}^{\infty}(u_s - u_{s+1})\right) \\
&= \prod_{s=1}^{t_0-1}(1-(1-p_{\text{partial}})(u_s - u_{s+1})) \cdot \exp\left(-(1-p_{\text{partial}})u_{t_0}\right).
\end{aligned}$$

Similarly, we have

$$\begin{aligned}
D &= \prod_{s=1}^{\infty}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \\
&= \prod_{s=1}^{t_0-1}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \cdot \prod_{s=t_0}^{\infty}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \\
&= \prod_{s=1}^{t_0-1}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \cdot (1-u_{t_0}+(1-p_{\text{partial}})u_{t_0+1}) \\
&\qquad \times \prod_{s=t_0+1}^{\infty}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \\
&\le \prod_{s=1}^{t_0-1}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \cdot (1-u_{t_0}+(1-p_{\text{partial}})u_{t_0+1}) \\
&\qquad \times \prod_{s=t_0+1}^{\infty}\exp(-u_s+(1-p_{\text{partial}})u_{s+1}) \\
&= \prod_{s=1}^{t_0-1}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \cdot (1-u_{t_0}+(1-p_{\text{partial}})u_{t_0+1}) \\
&\qquad \times \exp(-u_{t_0+1}) \prod_{s=t_0+1}^{\infty}\exp(-p_{\text{partial}}u_{s+1}) \\
&\le \prod_{s=1}^{t_0-1}(1-u_s+(1-p_{\text{partial}})u_{s+1}) \cdot (1-u_{t_0}+(1-p_{\text{partial}})u_{t_0+1}) \cdot \exp(-u_{t_0+1}).
\end{aligned}$$

Now let $f(u_{t_0+1}) := (1-u_{t_0}+(1-p_{\text{partial}})u_{t_0+1}) \cdot \exp(-u_{t_0+1})$ be a function of $u_{t_0+1} \in [0, u_{t_0}]$. Note that $f'(u_{t_0+1}) = \exp(-u_{t_0+1})((p_{\text{partial}}-1)u_{t_0+1}+u_{t_0}-p_{\text{partial}})$. If $u_{t_0} \le p_{\text{partial}}$, $f'(u_{t_0+1}) \le 0$ in the range $[0, u_{t_0}]$ and hence, $f(u_{t_0+1})$ achieves the maximum value of $f(0) = 1-u_{t_0}$. If $u_{t_0} > p_{\text{partial}}$, observe that $f'(0) > 0$ and f' is decreasing on $[0, u_{t_0}]$. Also, $f' = 0$ at $u_{t_0+1} = \frac{u_{t_0}-p_{\text{partial}}}{1-p_{\text{partial}}} \in [0, u_{t_0}]$. Thus, in this case, f gets the maximum value of

$$f\left(\frac{u_{t_0} - p_{\text{partial}}}{1 - p_{\text{partial}}}\right) = (1 - p_{\text{partial}}) \exp((u_{t_0} - p_{\text{partial}})/(p_{\text{partial}} - 1)).$$

We have showed the following bound.

Lemma 6.5.3. *For any fixed values of* $(p_{\text{full}}, p_{\text{partial}})$*,* $r_0, t_0 \geq 1$ *and* $1 \geq u_1 \geq \ldots \geq u_{t_0} \geq 0$*, we have*

$$AB + CD \leq F(r, u_1, \ldots, u_{t_0}),$$

where

$$F(r, u_1, \ldots, u_{t_0}) := \begin{cases} \left(1 - \frac{(1-p_{\text{full}})(1-u_1)}{r}\right)^r \alpha + \left(1 - p_{\text{full}} - \frac{(1-p_{\text{full}})(1-u_1)}{r}\right)^r \beta & \text{if } r < r_0 \\ e^{-(1-p_{\text{full}})(1-u_1)}\alpha + (1-p_{\text{full}})^{r_0} e^{u_1 - 1}\beta & \text{if } r = r_0 \end{cases},$$

$$\alpha := \prod_{s=1}^{t_0-1} (1 - (1 - p_{\text{partial}})(u_s - u_{s+1})) \cdot e^{-(1-p_{\text{partial}})u_{t_0}},$$

$$\beta := \prod_{s=1}^{t_0-1} (1 - u_s + (1 - p_{\text{partial}})u_{s+1}) \cdot \begin{cases} 1 - u_{t_0} & \text{if } u_{t_0} \leq p_{\text{partial}} \\ (1 - p_{\text{partial}}) e^{\frac{u_{t_0} - p_{\text{partial}}}{p_{\text{partial}} - 1}} & \text{if } u_{t_0} > p_{\text{partial}} \end{cases}$$

This implies that, given any $r_0, t_0 \geq 1$*, the (expected) approximation ratio of our algorithm is at most*

$$1 + \max_{\substack{r \in \{1,2,\ldots,r_0\} \\ 1 \geq u_1 \geq \ldots \geq u_{t_0} \geq 0}} \mathbf{E}_p \left[F(r, u_1, \ldots, u_{t_0})\right].$$

6.5.3 Computer-assisted analysis

By Lemma 6.5.3, to obtain the approximation ratio, we still need to solve the following (non-linear) optimization program:

$$\max_{\substack{r\in\{1,2,\ldots,r_0\}\\ 1\geq u_1\geq\ldots\geq u_{t_0}\geq 0}} \mathbf{E}_p[F(r,u_1,\ldots,u_{t_0})]. \tag{6.1}$$

We now choose the following parameters: $p_{\text{full}} := 0.43, p_{\text{partial}} := 0.055, r_0 := 5$, and $t_0 := 7$. Using Mathematica's NMaximize function to solve the above optimization program numerically, we get that F achieves a maximum value of 0.595057 at $r = 1, u_1 = 0.519441, u_2 = 0.399257, u_3 = 0.29198, u_4 = 0.198005, u_5 = 0.117735, u_6 = 0.0515877, u_7 = 0$. Unfortunately, since our problem is non-linear, Mathematica does not guarantee to return a global maximum. To get a more rigorous proof, we write another Java program to solve (6.1).

The high-level idea is to discretize the domain of F and use interval arithmetic to estimate an upper-bound of F on such small intervals. In particular, for a fixed value of $r \in \{1,\ldots,r_0\}$, we divide the interval $[0,1]$ (i.e., the domain of $u_1,\ldots,u_{t_0}$) into $M = 10000$ small segments of size $\epsilon = 1/M$. Recall that $F(u_1,\ldots,u_{t_0}) = S_0+S_1$ where both S_0 and S_1 can be written as a product (of t_0+1 factors) in which each factor only depends on two "consecutive" variables u_s and u_{s+1}. Note that $S_0, S_1 \in [0,1]$. For any vector u, let $S_0(s)$ and $S_1(s)$ denote the product of the first s factors of S_0 and S_1, respectively. To obtain an upper-bound on F, we can use the following dynamic programming algorithm. Let $G(s,i,j)$ be the maximum value of $S_1(s)$ for all $1 \geq u_1 \geq \ldots \geq u_s \geq 0$, where $0 \leq i,j \leq M-1$, $u_s \in [i/M,(i+1)/M]$, and $S_0(s) \in [j/M,(j+1)/M]$. Now one can easily compute G (recursively) in $O(M^3)$ time. By definition of G, we have

$$F \leq \max_{0 \leq i,j \leq M-1} \left\{ \frac{j+1}{M} + G(t_0,i,j) \right\}.$$

With a few other minor improvements, our program outputs the value of 0.59666972 after about one hour on a personal computer with a 2.3GHz Intel CPU and 16GB Memory. Thus, we proved the following theorem.

Theorem 6.5.1. *For $p_{\text{full}} = 0.43, p_{\text{partial}} = 0.055$,* ROUND2 *returns a solution $\mathcal{S}$ such that, for any $j \in V$, we have*

1. $\text{cost}(j) \leq 3R$,
2. $\mathbf{E}[\text{cost}(j)] \leq 1.59667R$.

The above result can be slightly improved by randomly picking $(p_{\text{full}}, p_{\text{partial}})$ before running ROUND2. The authors in [85] showed that if these two parameters are drawn as

$$(p_{\text{full}}, p_{\text{partial}}) = \begin{cases} (0.4525, 0) & \text{with probability } p = 0.773436 \\ (0.0480, 0.3950) & \text{with probability } 1-p \end{cases},$$

then $\mathbf{E}[\text{cost}(j)] \leq 1.592R$ for all $j \in V$.

Chapter 7: A Lottery Model for Center-type Problems With Outliers

7.1 Overview

In this chapter, we address several issues naturally arising in the standard center-type problems. First, it is not difficult to see that a few *outliers* (i.e., very distant clients) may result in a very large optimal radius in the center-type problems. This issue was raised by Charikar et. al. [60], who proposed a *robust* model in which we are given a parameter t and only need to serve t out of given n clients (i.e. $n - t$ *outliers* may be ignored in the solution). Here we consider three robust center-type problems: the Robust k-Center (RkCenter) problem, the Robust Knapsack Center (RKnapCenter) problem, and the Robust Matroid Center (RMatCenter) problem.

Formally, an instance $\mathcal{I}$ of the RkCenter problem consists of a set V of vertices, a metric distance d on V, an integer k, and an integer t. Let $n = |V|$ denote the number of vertices (clients). The goal is to choose a set $\mathcal{S} \subseteq V$ of centers (facilities) such that (i) $|\mathcal{S}| \leq k$, (ii) there is a set of *covered* vertices (clients) $\mathcal{C} \subseteq V$ of size at least t, and (iii) the objective function

$$R := \max_{j \in \mathcal{C}} \min_{i \in \mathcal{S}} d(i, j)$$

is minimized.

In the RKnapCenter problem, we are given a budget $B > 0$ instead of k. In addition, each vertex $i \in V$ has a weight $w_i \in \mathbb{R}_+$. The cardinality constraint (i) is replaced by the knapsack constraint: $\sum_{i \in \mathcal{S}} w_i \leq B$. Similarly, in the RMatCenter problem, the constraint (i) is replaced by a matroid constraint: $\mathcal{S}$ must be an independent set of a given matroid $\mathcal{M}$. Here we assume that we have access to the rank oracle of $\mathcal{M}$.

In [60], the authors introduced a greedy algorithm for the RkCenter problem that achieves an approximation ratio of 3. Recently, Chakrabarty et. al. [64] (independently) give a 2-approximation algorithm for this problem. Since the k-center problem is a special case of the RkCenter problem, this ratio is best possible unless P=NP.

The RKnapCenter problem was first studied by Chen et. al. [59]. In [59], the authors show that one can achieve an approximation ratio of 3 if allowed to slightly violate the knapsack constraint by a factor of $(1 + \epsilon)$. It is still unknown whether there exists a true approximation algorithm for this problem. The current inapproximability bound is still 3 due to the hardness of the Knapsack Center problem.

The current best approximation guarantee for the RMatCenter problem is 7 by Chen et. al. [59]. This problem has a hardness result of $(3 - \epsilon)$ via a reduction from the k-supplier problem.

From a practical viewpoint, unfairness arises inevitably in the robust model: some clients will always be considered as outliers and hence not covered within the guaranteed radius. To address this issue, we introduce a *lottery model* for these problems. The idea is to randomly pick a solution from a *public list* such that each client $j \in V$ is guaranteed to be covered with probability at least p_j, where $p_j \in [0, 1]$ is the success rate requested by j. In this paper, we introduce new approximation algorithms for these problems under this model. We also propose improved approximation algorithms for the RkCenter problem and the RMatCenter problem.

7.1.1 The Lottery Model

In this subsection, we formally define our lottery model for the above-mentioned problems. First, the *Fair* Robust k-Center (FRkCenter) problem is formulated as follows. Besides the parameters V, d, k and t, each vertex $j \in V$ has a "target" probability $p_j \in [0, 1]$. We are interested in the minimum radius R for which there exists a distribution $\mathcal{D}$ on subsets of V such that any set $\mathcal{S}$ drawn from $\mathcal{D}$ satisfies the following constraints:

Coverage constraint: $|\mathcal{C}| \geq t$, where $\mathcal{C}$ is the set of all clients in V that are within radius R from some center $\mathcal{S}$,

Fairness constraint: $\Pr[j \in \mathcal{C}] \geq p_j$ for all $j \in V$,

Cardinality constraint: $|\mathcal{S}| \leq k$.

Here we aim for a polynomial-time, randomized algorithm that can sample from $\mathcal{D}$. Note that the RkCenter is a special of this variant in which all p_j's are set to be zero.

The *Fair* Robust Knapsack Center (FRKnapCenter) problem and *Fair* Robust Matroid Center (FRMatCenter) problem are defined similarly except that we replace the cardinality constraint by a knapsack constraint and a matroid constraint, respectively. More formally, in the FRKnapCenter problem, we are given a budget $B \in \mathbb{R}^+$ and each vertex i has a weight $w_i \in \mathbb{R}^+$. We require the total weight of centers in $\mathcal{S}$ to be at most B with probability one. Similarly, in the FRMatCenter problem, we are given a matroid $\mathcal{M}$ and we require the solution $\mathcal{S}$ to be an independent set of $\mathcal{M}$ with probability one.

7.1.2 Our contributions and techniques

First of all, we give tight approximation algorithms for the RkCenter and RMatCenter problems.

Theorem 7.1.1. *There exist a 2-approximation algorithm for the RkCenter problem* [1] *and a 3-approximation algorithm for the RMatCenter problem.*

Our main results for the lottery model are summarized in the following theorems.

Theorem 7.1.2. *For any $\epsilon > 0$ and any instance $\mathcal{I} = (V, d, k, t, \vec{p})$ of the FRkCenter problem, there is a randomized algorithm $\mathcal{A}$ which can compute a random solution $\mathcal{S}$ such that*

- *$|\mathcal{S}| \leq k$ with probability one,*
- *$|\mathcal{C}| \geq (1-\epsilon)t$, where $\mathcal{C}$ is the set of all clients within radius $2R$ from some center in $\mathcal{S}$ and R is the optimal radius,*

[1]A 2-approximation algorithm has also been found independently by Chakrabarty et. al. [64], and in a private discussion between Marek Cygan and Samir Khuller. Our algorithm here is different from the algorithm in [64].

- $\Pr[j \in \mathcal{C}] \geq (1-\epsilon)p_j$ *for all* $j \in V$.

Theorem 7.1.3. *For any* $\epsilon > 0$ *and any instance* $\mathcal{I} = (V, d, w, B, t, \vec{p})$ *of the* FRKnapCenter *problem, there is a randomized algorithm* $\mathcal{A}$ *which can return random solution* $\mathcal{S}$ *such that*

- $\sum_{i \in \mathcal{S}} w_i \leq (1+\epsilon)B$ *with probability one,*
- $|\mathcal{C}| \geq t$*, where* $\mathcal{C}$ *is the set of vertices within distance* $3R$ *from some vertex in* $\mathcal{S}$*,*
- $\Pr[j \in \mathcal{C}] \geq p_j$ *for all* $j \in V$.

Finally, the FRMatCenter can be reduced to (randomly) rounding a point in a matroid intersection polytope. We design a rounding algorithm which can output a pseudo solution, consisting of a basis plus one extra center. By using a preprocessing step and solving a configuration LP, we can satisfy the matroid constraint exactly (respectively, knapsack constraint) while slightly violating the coverage and fairness constraints in the FRMatCenter (respectively, FRKnapCenter) problem.

Theorem 7.1.4. *For any* $\gamma > 0$ *and any instance* $\mathcal{I} = (V, d, \mathcal{M}, t, \vec{p})$ *of the* FRMatCenter *(respectively,* FRKnapCenter*) problem, there is a randomized algorithm* $\mathcal{A}$ *which can return a random solution* $\mathcal{S}$ *such that*

- $\mathcal{S}$ *is a basis of* $\mathcal{M}$ *with probability one, (respectively,* $w(\mathcal{S}) \leq B$ *with probability one)*

- *$|\mathcal{C}| \geq t - \gamma^2 n$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$,*

- *there exists a set $T \subseteq V$ of size at least $(1-\gamma)n$, which is deterministic, such that $\Pr[j \in \mathcal{C}] \geq p_j - \gamma$ for all $j \in T$.*

7.1.3 Organization

The rest of this chapter is organized as follows. In the next section, we review some basic properties of matroids and discuss a filtering algorithm which is used in later algorithms. Then we develop rounding algorithms for the FRkCenter, FRKnapCenter, and FRMatCenter in the next sections.

7.2 Preliminaries

7.2.1 Matroid polytopes

We first review a few basic facts about matroid polytopes. For any vector z and set S, we let $z(S)$ denote the sum $\sum_{i \in S} z_i$. Let $\mathcal{M}$ be any matroid on the ground set Ω and $r_{\mathcal{M}}$ be its rank function. The matroid base polytope of $\mathcal{M}$ is defined by

$$\mathcal{P}_{\mathcal{M}} := \left\{ x \in \mathbb{R}^{\Omega} : x(S) \leq r_{\mathcal{M}}(S) \ \ \forall S \subseteq \Omega; \quad x(\Omega) = r_{\mathcal{M}}(\Omega); \quad x_i \geq 0 \ \ \forall i \in \Omega \right\}.$$

Definition 7.2.1. *Suppose $Ax \leq b$ is a valid inequality of $\mathcal{P}_{\mathcal{M}}$. A face D of $\mathcal{P}_{\mathcal{M}}$ (corresponding to this valid inequality) is the set $D := \{x \in \mathcal{P}_{\mathcal{M}} : Ax = b\}$.*

The following theorem gives a characterization for any face of $\mathcal{P}_{\mathcal{M}}$ (See, e.g., [88]).

Theorem 7.2.1. *Let D be any face of $\mathcal{P}_{\mathcal{M}}$. Then it can be characterized by*

$$D = \left\{ x \in \mathbb{R}^{\Omega} : x(S) = r_{\mathcal{M}}(S) \;\; \forall S \in \mathcal{L}; \quad x_i = 0 \;\; \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}} \right\},$$

where $J \subseteq \Omega$ and $\mathcal{L}$ is a chain family of sets: $L_1 \subset L_2 \subset \ldots \subset L_m$. Moreover, it is sufficient to choose $\mathcal{L}$ as any maximal chain $L_1 \subset L_2 \subset \ldots \subset L_m$ such that $x(L_i) = r_{\mathcal{M}}(L_i)$ for all $i = 1, 2, \ldots, m$.

Proposition 7.2.1. *Let $x \in \mathcal{P}_{\mathcal{M}}$ be any point and I be the set of all tight constraints of $\mathcal{P}_{\mathcal{M}}$ on x. Suppose D is the face with respect to I. Then one can compute a chain family $\mathcal{L}$ for D as in Theorem 7.2.1 in polynomial time.*

Proof. Recall that $r_{\mathcal{M}}$ is a submodular function. Then observe that the function $r'_{\mathcal{M}}(S) = r_{\mathcal{M}}(S) - x(S)$ for $S \subseteq \Omega$ is also submodular. It is well-known that submodular minimization can be done in polynomial time. We solve the following optimization problem: $\min \{r'_{\mathcal{M}}(S) : S \subseteq \Omega\}$. If there are multiple solutions, we let S_0 be any solution of minimal size. (This can be done easily, say, by trying to drop each item from the current solution and resolving the program.) We add S_0 to our chain. Then we find some *minimal* superset S_1 of S_0 such that $r'_{\mathcal{M}}(S_1) = 0$, add S_1 to our chain, and repeat the process. □

Corollary 7.2.1. *Let D be any face of $\mathcal{P}_{\mathcal{M}}$. Then it can be characterized by*

$$D = \left\{x \in \mathbb{R}^{\Omega} : x(S) = b_S \ \ \forall S \in \emptyset; \quad x_i = 0 \ \ \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}}\right\},$$

where $J \subseteq \Omega$ and $\emptyset$ is a family of pairwise disjoint sets: $O_1, O_2, \ldots, O_m$, and $b_{O_1}, \ldots, b_{O_m}$ are some constants.

Proof. By Theorem 7.2.1, we have that

$$D = \left\{x \in \mathbb{R}^{\Omega} : x(S) = r_{\mathcal{M}}(S) \ \ \forall S \in \mathcal{L}; \quad x_i = 0 \ \ \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}}\right\},$$

where $J \subseteq \Omega$ and $\mathcal{L}$ is the chain: $L_1 \subset L_2 \subset \ldots \subset L_m$. Now let us define $O_1 := L_1, O_2 := L_2 \setminus L_1, O_3 := L_3 \setminus L_2, \ldots, O_m := L_m \setminus L_{m-1}$, and $b_{O_1} := r_{\mathcal{M}}(L_1), b_{O_2} := r_{\mathcal{M}}(L_2) - r_{\mathcal{M}}(L_1), \ldots, b_{O_m} := r_{\mathcal{M}}(L_m) - r_{\mathcal{M}}(L_{m-1})$. It is not difficult to verify that

$$D = \left\{x \in \mathbb{R}^{\Omega} : x(S) = b_S \ \ \forall S \in \emptyset; \quad x_i = 0 \ \ \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}}\right\}.$$

□

7.2.2 Filtering algorithm

All algorithms in this chapter are based on rounding an LP solution. In general, for each vertex $i \in V$, we have a variable $y_i \in [0, 1]$ which represents the probability that we want to pick i in our solution. (In the standard model, y_i is the "extent" that i is opened.) In addition, for each pair of $i, j \in V$, we have a variable $x_{ij} \in [0, 1]$

which represents the probability that j is connected to i.

Note that in all center-type problems, the optimal radius R is always the distance between two vertices. Therefore, we can always "guess" the value of R in $O(n^2)$ time. WLOG, we may assume that we know the correct value of R. For any $j \in V$, we let $F_j := \{i \in V : d(i,j) \leq R \wedge x_{ij} > 0\}$ and $s_j := \sum_{i \in V: d(i,j) \leq R} x_{ij}$. We shall refer to F_j as a cluster with cluster center j. Depending on a specific problem, we may have different constraints on x_{ij}'s and y_i's. In general, the following constraints are valid in most of the problems here:

$$\sum_{j \in V} \sum_{i \in V: d(i,j) \leq R} x_{ij} \geq t, \tag{7.1}$$

$$\sum_{i \in V: d(i,j) \leq R} x_{ij} \leq 1, \quad \forall j \in V, \tag{7.2}$$

$$x_{ij} \leq y_i, \quad \forall i, j \in V, \tag{7.3}$$

$$y_i, x_{ij} \geq 0, \quad \forall i, j \in V. \tag{7.4}$$

For the *fair* variants, we may also require that

$$\sum_{i \in V: d(i,j) \leq R} x_{ij} \geq p_j, \quad \forall j \in V. \tag{7.5}$$

Constraint (7.1) says that at least t vertices should be covered. Constraint (7.2) ensures that each vertex is only connected to at most one center. Constraint (7.3) means vertex j can only connect to center i if it is open. Constraint (7.5) says that the total probability of j being connected should be at least p_j. By constraints

(7.2) and (7.3), we have $y(F_j) \leq 1$.

The first step of all algorithms in this chapter is to use the following *filtering* algorithm to obtain a maximal collection of disjoint clusters. The algorithm will return the set V' of cluster centers of the chosen clusters. In the process, we also keep track of the number c_j of other clusters removed by F_j for each $j \in V'$.

Algorithm 21 RFILTERING (x, y)

```
1: $V' \leftarrow \emptyset$
2: **for each** unmarked cluster $F_j$ in **decreasing order** of $s_j = \sum_{i \in V : d(i,j) \leq R} x_{ij}$ **do**
3:     $V' \leftarrow V' \cup \{j\}$
4:     Set all unmarked clusters $F_k$ (including $F_j$ itself) s.t. $F_k \cap F_j \neq \emptyset$ as marked.
5:     Let $c_j$ be the number of marked clusters in this step.
6: $\vec{c} \leftarrow (c_j : j \in V')$
7: **return** $(V', \vec{c})$
```

7.3 The k-center problems with outliers

In this section, we first give a simple 2-approximation algorithm for the RkCenter problem. Then, we give an approximation algorithm for the FRkCenter problem, proving Theorem 7.1.2.

7.3.1 The robust k-center problem

Suppose $\mathcal{I} = (V, d, k, t)$ is an instance the RkCenter problem with the optimal radius R. Consider the polytope $\mathcal{P}_{\mathsf{RkCenter}}$ containing points (x, y) satisfying constraints (7.1)–(7.4), and the cardinality constraint:

$$\sum_{i \in V} y_i \le k. \tag{7.6}$$

Since R is the optimal radius, it is not difficult to check that $\mathcal{P}_{\mathsf{RkCenter}} \neq \emptyset$. Let us pick any fractional solution $(x, y) \in \mathcal{P}_{\mathsf{RkCenter}}$. The next step is to round (x, y) into an integral solution using the following simple algorithm.

Algorithm 22 RkCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: $\mathcal{S} \leftarrow$ the top $k$ vertices $i \in V'$ with highest value of $c_i$.
3: return $\mathcal{S}$
```

Analysis. By construction, the algorithm returns a set $\mathcal{S}$ of k open centers. Note that, for each $i \in \mathcal{S}$, c_i is the number of distinct clients within radius $2R$ from i. Thus, it suffices to show that $\sum_{i \in \mathcal{S}} c_i \ge t$. By inequality (7.2), we have that $s_j \le 1$ for all $j \in V'$. Thus,

$$\sum_{i \in V'} c_i s_i \ge \sum_{i \in V} s_i \ge t,$$

where the first inequality is due to the greedy choice of vertices in V' and the second inequality follows by (7.1). Now recall that the clusters whose centers in V' are pairwise disjoint. By constraint (7.6), we have

$$\sum_{i \in V'} s_i \le \sum_{i \in V'} y(F_i) \le \sum_{i \in V} y_i \le k.$$

It follows by the choice of $\mathcal{S}$ that $\sum_{i \in \mathcal{S}} c_i \ge t$. This concludes the first part of

Theorem 7.1.1.

7.3.2 The fair robust k-center problem

Assume $\mathcal{I} = (V, d, k, t, \vec{p})$ be an instance of the $\mathsf{FRkCenter}$ problem with the optimal radius R. Fix any $\epsilon > 0$. If $k \le 2/\epsilon$, then we can generate all possible $O\left(n^{1/\epsilon}\right)$ solutions and then solve an LP to obtain the corresponding marginal probabilities. So the problem can be solved easily in this case. We will assume that $k \ge 2/\epsilon$ for the rest of this section. Consider the polytope $\mathcal{P}_{\mathsf{FRkCenter}}$ containing points (x, y) satisfying constraints (7.1)–(7.4), the fairness constraint (7.5), and the cardinality constraint (7.6). We now show that $\mathcal{P}_{\mathsf{FRkCenter}}$ is actually a valid relaxation polytope.

Proposition 7.3.1. *We have that $\mathcal{P}_{\mathsf{FRkCenter}} \neq \emptyset$.*

Proof. It suffices to point out a solution $(x, y) \in \mathcal{P}_{\mathsf{FRkCenter}}$. Since R is the optimal radius, there exists a distribution $\mathcal{D}$ satisfying the coverage, fairness, and cardinality constraints. Suppose $\mathcal{S}$ is sampled from $\mathcal{D}$ and $\mathcal{C}$ is the set of all clients in V that are within radius R from some center $\mathcal{S}$. We now set $y_i := \Pr[i \in \mathcal{S}]$ for all $i \in V$. Since $|\mathcal{S}| \le k$ with probability one, we have $\sum_{i \in V} y_i = \mathbf{E}[|\mathcal{S}|] \le k$, and hence constraint (7.6) is valid.

We construct the assignment variable x as follows. For each $j \in V$, let $S_j := \{i : d(i, j) \le R\}$, $z_j := 0$. Then for each $i \in S_j$, set $x_{ij} := \min\{y_i, 1 - z_j\}$ and update $z_j := z_j + x_{ij}$. We repeat the process for all vertices in S_j. It is not hard to see that inequalities (7.2) and (7.3) hold by this construction. Now let us fix any $j \in V$. By fairness guarantee of $\mathcal{D}$ and the union bound, we have

$$p_j \leq \Pr[j \in \mathcal{C}] \leq \sum_{i \in V : d(i,j) \leq R} y_i.$$

Thus, by construction of x, we have

$$\sum_{i \in V : d(i,j) \leq R} x_{ij} \geq \Pr[j \in \mathcal{C}] \geq p_j,$$

and hence inequality (7.5) is satisfied. Finally, we have

$$\mathbf{E}[|\mathcal{C}|] = \sum_{j \in V} \Pr[j \in \mathcal{C}] \leq \sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x_{ij}.$$

Since $|\mathcal{C}| \geq t$ with probability one, $\mathbf{E}[|\mathcal{C}|] \geq t$, implying that inequality (7.1) holds.

$\square$

Fix any small parameter $\epsilon > 0$. Our algorithm is as follows.

Algorithm 23 FRkCenterRound (ϵ, x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$.
2: for each $j \in V'$ do
3:     $y'_j \leftarrow (1 - \epsilon) \sum_{i \in F_j} x_{ij}$
4: while $y'$ still contains $\geq 3$ fractional values in $(0, 1)$ do
5:     Let $\delta \in \mathbb{R}^{V'}, \delta \neq 0$ be such that $\delta_i = 0 \ \forall i \in V' : y'_i \in \{0, 1\}$, $\delta(V') = 0$, and
       $\vec{c} \cdot \delta = 0$.
6:     Choose scaling factors $a, b > 0$ such that
         • $y' + a\delta \in [0, 1]^{V'}$ and $y' - b\delta \in [0, 1]^{V'}$
         • there is at least one new entry of $y' + a\delta$ which is equal to zero or one
         • there is at least one new entry of $y' - b\delta$ which is equal to zero or one
7:     With probability $\frac{b}{a+b}$, update $y' \leftarrow y' + a\delta$; else, update $y' \leftarrow y' - b\delta$.
8: return $\mathcal{S} = \{i \in V : y'_i > 0\}$.
```

Analysis. First, note that one can find such a vector δ in line 5 as the system of $\delta(V') = 0$ and $\vec{c} \cdot \delta = 0$ consists of two constraints and at least 3 variables (and hence is underdetermined.) By construction, at least one more fractional variable becomes rounded after each iteration. Thus, the algorithm terminates after $O(n)$ rounds. Let $\mathcal{S}$ denote the (random) solution returned by FRkCenterRound and $\mathcal{C}$ be the set of all clients within radius $3R$ from some center in $\mathcal{S}$. Theorem 7.1.2 can be verified by the following propositions.

Proposition 7.3.2. *$|\mathcal{S}| \leq k$ with probability one.*

Proof. By definition of y' at line 2 of FRkCenterRound, we have

$$\begin{aligned} y'(V') = \sum_{j \in V'} y'_j &= (1-\epsilon) \sum_{j \in V'} \sum_{i \in F_j} x_{ij} \\ &\leq (1-\epsilon) \sum_{j \in V'} \sum_{i \in F_j} y_i \leq (1-\epsilon)k \leq k-2, \end{aligned}$$

since $k \geq 2/\epsilon$. Note that the sum $y'(V')$ is never changed in the while loop (lines $4 \ldots 7$) because $\delta(V') = 0$. Then the final vector y' contains at most two fractional values at the end of the while loop. By rounding these two values to one, the size of $\mathcal{S}$ is indeed at most k. □

Proposition 7.3.3. *$|\mathcal{C}| \geq (1-\epsilon)t$ with probability one.*

Proof. At the beginning of the while loop, we have

$$\vec{c} \cdot y' = \sum_{j \in V'} c_j y'_j(F_j) = (1-\epsilon) \sum_{j \in V'} c_j s_j \geq (1-\epsilon) \sum_{j \in V} s_j \geq (1-\epsilon)t.$$

Again, the quantity $\vec{c} \cdot y'$ is unchanged in the while loop because $\vec{c} \cdot \delta = 0$ implies that $\vec{c} \cdot (y' + a\delta) = \vec{c} \cdot y'$ and $\vec{c} \cdot (y' - b\delta) = \vec{c} \cdot y'$ in each iteration. Note that if $y' \in \{0,1\}^{V'}$, then $\vec{c} \cdot y'$ is the number of clients within radius $2R$ from some center i such that $y'_i = 1$. Basically, we round the two remaining fractional values of y' to one in line 8; and hence, the dot product should be still at least $(1-\epsilon)t$. $\square$

Proposition 7.3.4. $\Pr[j \in \mathcal{C}] \geq (1-\epsilon)p_j$ *for all* $j \in V$.

Proof. Fix any $j \in V$. The algorithm RFiltering guarantees that there exists $k \in V'$ such that $F_j \cap F_k \neq \emptyset$ and $s_k \geq s_j$. Now we claim that $\mathbf{E}[y'_k] = y'_k$. This is because the expected value of y'_k does not change after any single iteration:

$$\mathbf{E}[y'_k] = (y'_k + a\delta)\frac{b}{a+b} + (y'_k - b\delta)\frac{a}{a+b} = y'_k.$$

Then we have that

$$\Pr[k \in \mathcal{S}] = \Pr[y'_k > 0] \geq \Pr[y'_k = 1] = \mathbf{E}[y'_k] = y'_k = (1-\epsilon)s_k.$$

Therefore,

$$\Pr[j \in \mathcal{C}] \geq \Pr[k \in \mathcal{S}] \geq (1-\epsilon)s_k \geq (1-\epsilon)s_j \geq (1-\epsilon)p_j,$$

by constraint (7.5).

$\square$

7.4 The Knapsack Center problems with outliers

We study the RKnapCenter and FRKnapCenter problems in this section. Recall that in these problems, each vertex has a weight and we want to make sure that the total weight of the chosen centers does not exceed a given budget B. We first give a 3-approximation algorithm for the RKnapCenter problem that slightly violates the knapsack constraint. Although this is not better than the known result by [59], both our algorithm and analysis here are more natural and simpler. It serves as a starting point for the next results. For the FRKnapCenter, we show that it is possible to satisfy the knapsack constraint exactly with small violations in the coverage and fairness constraints.

7.4.1 The robust knapsack center problem

Suppose $\mathcal{I} = (V, d, w, B, t)$ is an instance the RKnapCenter problem with the optimal radius R. Consider the polytope $\mathcal{P}_{\mathsf{RKnapCenter}}$ containing points (x, y) satisfying constraints (7.1)–(7.4), and the knapsack constraint:

$$\sum_{i \in V} w_i y_i \leq B. \tag{7.7}$$

Again, it is not difficult to check that $\mathcal{P}_{\mathsf{RKnapCenter}} \neq \emptyset$. Let us pick any fractional solution $(x, y) \in \mathcal{P}_{\mathsf{RKnapCenter}}$. Our pseudo-approximation algorithm to round (x, y) is as follows.

Analysis. We first claim that $\mathcal{P}' \neq \emptyset$ which implies that the extreme point Y of $\mathcal{P}'$

Algorithm 24 RKnapCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: For each $i \in V'$, let $v_i \leftarrow \arg\min_{j \in F_i}\{w_j\}$ be the vertex with smallest weight in $F_i$
3: Let $\mathcal{P}' := \left\{z \in [0,1]^{V'} : \sum_{i \in V'} c_i z_i \geq t \quad \wedge \quad \sum_{i \in V'} w_{v_i} z_i \leq B\right\}$
4: Compute an extreme point $Y$ of $\mathcal{P}'$
5: return $\mathcal{S} = \{v_i : i \in V,\ Y_i > 0\}$
```

(in line 4) does exist. To see this, let $z_i := s_i$ for all $i \in V'$. Then we have

$$\sum_{i \in V'} c_i z_i = \sum_{i \in V'} c_i s_i \geq \sum_{i \in V} s_i \geq t.$$

Also,

$$\begin{aligned}
\sum_{i \in V'} w_{v_i} z_i &= \sum_{i \in V'} w_{v_i} s_i \\
&= \sum_{i \in V'} w_{v_i} \sum_{j \in F_i} x_{ji} \\
&\leq \sum_{i \in V'} w_{v_i} \sum_{j \in F_i} y_j \\
&\leq \sum_{i \in V'} \sum_{j \in F_i} w_j y_j \leq \sum_{i \in V} w_i y_i \leq B.
\end{aligned}$$

All the inequalities follow from LP constraints and definitions of s_i, c_i, and v_i. Thus, $z \in \mathcal{P}'$, implying that $\mathcal{P}' \neq \emptyset$.

Proposition 7.4.1. RKnapCenterRound *returns a solution $\mathcal{S}$ such that $w(\mathcal{S}) \leq B + 2w_{max}$ and $|\mathcal{C}| \geq t$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$ and w_{max} is the maximum weight of any vertex in V.*

Proof. First, observe that any extreme point of $\mathcal{P}'$ has at most 2 fractional values.

(In the worst case, an extreme point z is fully determined by $|V'|-2$ tight constraints of the form $z_i = 0$ or $z_i = 1$, $\sum_{i \in V'} c_i z_i = t$, and $\sum_{i \in V'} w_{v_i} z_i = B$.) By construction of $\mathcal{S}$, we may also pick at most 2 vertices i^*, i^{**} that $Y_{i^*}, Y_{i^{**}}$ are fractional. Thus,

$$w(\mathcal{S}) = \sum_{i \in \mathcal{S} \setminus \{i^*, i^{**}\}} w_{v_i} Y_i + w_{i^*} + w_{i^{**}} \leq B + 2w_{\max}.$$

Recall that for each $i \in V'$, there are c_i clients at distance $\leq 2R$ from i (and each client is counted only one time.) By triangle inequality, these clients are within distance $3R$ from v_i. Thus, $\mathcal{S}$ will cover at least

$$\sum_{i \in \mathcal{S} \setminus \{i^*, i^{**}\}} c_i Y_i + c_{i^*} + c_{i^{**}} \geq \sum_{i \in \mathcal{S}} c_i Y_i \geq t,$$

clients within radius $3R$.

□

7.4.2 The fair robust knapsack center problem

In this section, we will first consider a simple algorithm that only violates the knapsack constraint by two times the maximum weight of any vertex. Then using a configuration polytope to "condition" on the set of "big" vertices, we show that it is possible to either violate the budget by $(1 + \epsilon)$ or to preserve the knapsack constraint while slightly violating the coverage and fairness constraints.

7.4.2.1 Basic algorithm

Suppose $\mathcal{I} = (V, d, w, B, t, \vec{p})$ is an instance the FRKnapCenter problem with the optimal radius R. Consider the polytope $\mathcal{P}_{\mathsf{FRKnapCenter}}$ containing points (x, y) satisfying constraints (7.1)–(7.4), the fairness constraint (7.5), and the knapsack constraint (7.7). The proof that $\mathcal{P}_{\mathsf{FRKnapCenter}} \neq \emptyset$ is very similar to that of Proposition 7.3.1 and is omitted here.

The following algorithm is a randomized version of RKnapCenterRound.

Algorithm 25 BasicFRKnapCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: For each $i \in V'$ let $v_i := \arg\min_{j \in F_i}\{w_j\}$ be the vertex with smallest weight in
   $F_i$
3: Let $\mathcal{P}' := \left\{z \in [0,1]^{V'} : \sum_{i \in V'} c_i z_i \geq t \quad \wedge \quad \sum_{i \in V'} w_{v_i} z_i \leq B\right\}$
4: Let $z_i \leftarrow s_i$ for all $i \in V'$. Write $z$ as a convex combination of extreme points
   $z^{(1)}, \ldots, z^{(n+1)}$ of $\mathcal{P}'$:
         $z = p_1 z^{(1)} + \ldots + p_{n+1} z^{(n+1)},$
   where $\sum_\ell p_\ell = 1$ and $p_\ell \geq 0$ for all $\ell \in [n+1]$.
5: Randomly choose $Y \leftarrow z_\ell$ with probability $p_\ell$.
6: return  $\mathcal{S} = \{v_i : i \in V,\ Y_i > 0\}$
```

Analysis. It is not hard to verify that $\mathcal{P}' \neq \emptyset$ (see the analysis in Section 7.4.1). This means that the decomposition at line 4 can be done.

Proposition 7.4.2. *The algorithm* BasicFRKnapCenterRound *returns a random solution $\mathcal{S}$ such that $w(\mathcal{S}) \leq B + 2w_{max}$, $|\mathcal{C}| \geq t$, and* $\Pr[j \in \mathcal{C}] \geq p_j$ *for all $j \in V$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$ and w_{max} is the maximum weight of any vertex in V.*

Proof. With similar arguments as in the proof of Proposition 7.4.1, we have that $w(\mathcal{S}) \leq B + 2w_{\max}$ and $|\mathcal{C}| \geq t$. To obtain the fairness guarantee, observe that

$v_i \in \mathcal{S}$ with probability at least $z_i = s_i$. For any $j \in V$, let $k \in V'$ be the vertex that removed j in the filtering step. We have

$$\Pr[j \in \mathcal{C}] \geq \Pr[v_k \in \mathcal{S}] \geq s_k \geq s_j \geq p_j,$$

where the penultimate inequality is due to our greedy choice of k in RFiltering.

□

7.4.2.2 An algorithm slightly violating the budget constraint

Fix a small parameter $\epsilon > 0$. A vertex i is said to be *big* iff $w_i > \epsilon B$. Then there can be at most $1/\epsilon$ big vertices in a solution. Let $\mathcal{U}$ denote the collection of all possible sets of big vertices. We have that $|\mathcal{U}| \leq n^{O(1/\epsilon)}$. Consider the *configuration* polytope $\mathcal{P}_{\textsf{config1}}$ containing points (x, y, q) with the following constraints:

$$\begin{cases} \sum_{U \in \mathcal{U}} q_U = 1 & \\ \sum_{i \in V : d(i,j) \le R} x_{ij}^U \le q_U & \forall j \in V, U \in \mathcal{U} \\ \sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \le R} x_{ij}^U \ge p_j & \forall j \in V \\ x_{ij}^U \le y_i^U & \forall i, j \in V, U \in \mathcal{U} \\ \sum_{i \in V} w_i y_i^U \le q_U B & \forall U \in \mathcal{U} \\ \sum_{j \in V} \sum_{i \in V : d(i,j) \le R} x_{ij}^U \ge q_U t & \\ y_i^U = 1 & \forall U \in \mathcal{U}, i \in U \\ y_i^U = 0 & \forall U \in \mathcal{U}, i \in V \setminus U, w_i > 1/\epsilon \\ x_{ij}^U, y_i^U, q_U \ge 0 & \forall i, j \in V, U \in \mathcal{U} \end{cases}$$

We first claim that $\mathcal{P}_{\textsf{config1}}$ is a valid relaxation polytope for the problem.

Proposition 7.4.3. *We have that $\mathcal{P}_{\textsf{config1}} \neq \emptyset$.*

Proof. Fix any optimal distribution $\mathcal{D}$. Suppose $\mathcal{S}$ is sampled from $\mathcal{D}$. For any $U \in \mathcal{U}$, set q_U to be the probability that $U \subseteq \mathcal{S}$ and $\mathcal{S} \setminus U$ contains no big vertex. Then it is clear that $\sum_{U \in \mathcal{U}} q_U = 1$. Let $\mathcal{E}(U)$ denote this event. Let x_{ij}^U be probability of the joint event: $\mathcal{E}(U)$ and j is connected to i. Finally, let y_i^U be the probability of the joint event: $\mathcal{E}(U)$ and $i \in \mathcal{S}$.

Now observe that

$$\begin{aligned} q_U &= \Pr[\mathcal{E}(U)] \\ &\geq \Pr[\mathcal{E}(U) \wedge j \text{ is connected}] \\ &= \sum_{i \in V : d(i,j) \leq R} \Pr[j \text{ is connected to } i \wedge \mathcal{E}(U)] \\ &= \sum_{i \in V : d(i,j) \leq R} x_{ij}^U. \end{aligned}$$

Similarly,

$$\begin{aligned} p_j &\leq \Pr[j \text{ is connected}] \\ &= \sum_{U \in \mathcal{U}} \Pr[j \text{ is connected} \wedge \mathcal{E}(U)] \\ &= \sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \leq R} \Pr[j \text{ is connected to } i \wedge \mathcal{E}(U)] \\ &= \sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \leq R} x_{ij}^U. \end{aligned}$$

Note that x_{ij}^U / q_U and y_i^U / q_U are the probabilities that j is connected to i and $i \in \mathcal{S}$ conditioned on $\mathcal{E}(U)$, respectively. Since the number of connected clients is at least t with probability one, we have

$$\begin{aligned}
t &\geq \mathbf{E}[\#\text{ connected clients}|\mathcal{E}(U)] \\
&= \sum_{j \in V} \Pr[j \text{ is served}|\mathcal{E}(U)] \\
&= \sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} \Pr[j \text{ is connected to } i|\mathcal{E}(U)] \\
&= \sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x_{ij}^U / q_U.
\end{aligned}$$

Similarly, since $w(\mathcal{S}) \leq B$ with probability one, we have

$$B \geq \mathbf{E}[w(\mathcal{S})|U] = \sum_{i \in V} w_i (y_i^U / q_U).$$

The other constraints can be verified easily. We conclude that $(x, y, q) \in \mathcal{P}_{\mathsf{config1}}$ and $\mathcal{P}_{\mathsf{config1}} \neq \emptyset$. □

Next, let us pick any $(x, y, q) \in \mathcal{P}_{\mathsf{config1}}$ and use the following algorithm to round it.

Algorithm 26 FRKNAPCENTERROUND1 (x, y, q)

```
1: Randomly pick a set $U \in \mathcal{U}$ with probability $q_U$
2: Let $x'_{ij} \leftarrow x_{ij}^U / q_U$ and $y'_i \leftarrow \min\{y_i^U / q_U, 1\}$
3: return $\mathcal{S}$ = BASICRFKNAPCENTERROUND $(x', y')$
```

We are now ready to prove Theorem 7.1.3.

Proof of Theorem 7.1.3. We will show that FRKNAPCENTERROUND1 will return a solution $\mathcal{S}$ with properties in Theorem 7.1.3. Let $\mathcal{E}(U)$ denote the event that $U \in \mathcal{U}$ is picked in the algorithm. Note that (x', y') satisfies the following constraints:

$$\sum_{j \in V} \sum_{i \in V : d(i,j) \le R} x'_{ij} \ge t,$$

$$\sum_{i \in V : d(i,j) \le R} x'_{ij} \le 1, \quad \forall j \in V,$$

$$\sum_{i \in V : d(i,j) \le R} x'_{ij} = \sum_{i \in V : d(i,j) \le R} x_{ij}/q_U, \quad \forall j \in V,$$

$$x'_{ij} \le y'_i, \quad \forall i, j \in V,$$

$$\sum_{i \in V} w_i y'_i \le B.$$

Moreover, $y'_i = 1$ for all $i \in U$ and $y'_i = 0$ for all $i \in V \setminus U$ and $w_i > \epsilon B$. Thus, the two extra fractional vertices opened by BasicFRKnapCenterRound will have weight at most ϵB. By Proposition 7.4.2, we have $w(\mathcal{S}) \le B + 2\epsilon B = (1+2\epsilon)B$. Moreover, conditioned on U, we have

$$\Pr[j \in \mathcal{C} | \mathcal{E}(U)] \ge \sum_{i \in V : d(i,j) \le R} x'_{ij} = \sum_{i \in V : d(i,j) \le R} x_{ij}/q_U.$$

Thus, by definition of $\mathcal{P}_{\mathsf{config1}}$ and our construction of $\mathcal{S}$, we get

$$\begin{aligned} \Pr[j \in \mathcal{C}] &= \sum_{U \in \mathcal{U}} \Pr[j \in \mathcal{C} | \mathcal{E}(U)] \Pr[\mathcal{E}(U)] \\ &\ge \sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \le R} x_{ij} \\ &\ge p_j. \end{aligned}$$

$\square$

7.4.2.3 An algorithm that satisfies the knapsack constraint exactly

Let $\epsilon > 0$ a small parameter to be determined. Let $\mathcal{U}$ denote the collection of all possible sets of verticies with size at most $\lceil 1/\epsilon \rceil$. We have that $|\mathcal{U}| \leq n^{O(1/\epsilon)}$. Suppose R is the optimal radius to our instance. Given a set $U \in \mathcal{U}$, we say that vertex $j \in V$ is *blue* if there exists $i \in U$ such that $d(i,j) \leq 3R$. Otherwise, vertex i is said to be *red*. For any $i \in V$, let $\mathrm{RBall}(i, U, R)$ denote the set of red vertices within radius $3R$ from i:

$$\mathrm{RBall}(i, U, R) := \{ j \in V : (d(i,j) \leq 3R \ \wedge \ \nexists k \in U : d(k,j) \leq 3R) \}.$$

Consider the *configuration* polytope $\mathcal{P}_{\mathsf{config2}}$ containing points (x, y, q) with the following constraints:

$$\begin{cases}
\sum_{U\in\mathcal{U}} q_U = 1 & \\
\sum_{i\in V: d(i,j)\le R} x_{ij}^U \le q_U & \forall j\in V, U\in\mathcal{U} \\
\sum_{U\in\mathcal{U}} \sum_{i\in V: d(i,j)\le R} x_{ij}^U \ge p_j & \forall j\in V \\
x_{ij}^U \le y_i^U & \forall i,j\in V, U\in\mathcal{U} \\
\sum_{i\in V} w_i y_i^U \le q_U B & \forall U\in\mathcal{U} \\
\sum_{j\in V} \sum_{i\in V: d(i,j)\le R} x_{ij}^U \ge q_U t & \\
y_i^U = 1 & \forall U\in\mathcal{U}, i\in U \\
y_i^U = 0 & \forall U\in\mathcal{U}, i\in V\setminus U, |\mathrm{RBall}(i,U,R)|\ge \epsilon n \\
x_{ij}^U, y_i^U, q_U \ge 0 & \forall i,j\in V, U\in\mathcal{U}
\end{cases}$$

We first claim that $\mathcal{P}_{\mathsf{config2}}$ is a valid relaxation polytope for the problem.

Proposition 7.4.4. *We have that $\mathcal{P}_{\mathsf{config2}} \neq \emptyset$.*

Proof. Suppose $\mathcal{S}$ is a solution drawn from the optimal distribution $\mathcal{D}$. We now compute a subset $U_{\mathcal{S}}$ of $\mathcal{S}$ using the following procedure. Initially, set $U_{\mathcal{S}} := \emptyset$ and all vertices in V are marked as red. While there exists $i\in\mathcal{S}$ such that there are at least ϵn red vertices within radius $3R$ from i (i.e., $|\mathrm{RBall}(i, U_{\mathcal{S}}, R)| \ge \epsilon n$), pick any such vertex i, breaking ties by choosing the one with smallest index. Then we set $U_{\mathcal{S}} := U_{\mathcal{S}} \cup \{i\}$, mark all vertices within radius $3R$ from i as blue, and repeat the process.

Note that for all $i \in \mathcal{S} \setminus U_{\mathcal{S}}$, we have $|\text{RBall}(i, U_{\mathcal{S}}, R)| < \epsilon n$ by the condition of the while-loop. Moreover, we have $|U_{\mathcal{S}}| \leq \lceil 1/\epsilon \rceil$ so $U_{\mathcal{S}} \in \mathcal{U}$. (Suppose $|U_{\mathcal{S}}| > 1/\epsilon$. For each $i \in U_{\mathcal{S}}$, there are at least ϵn red vertices turned into blue by i in the procedure. This implies that there are more than $(1/\epsilon) \times \epsilon n = n$ vertices, which is a contradiction.)

Now for any $U \in \mathcal{U}$, we set $q_U := \Pr[U_{\mathcal{S}} = U]$. Let x_{ij}^U be probability of the joint event: $U_{\mathcal{S}} = U$ and j is connected to i. Finally, let y_i^U be the probability of the joint event: $U_{\mathcal{S}} = U$ and $i \in \mathcal{S}$. Then it is clear that $\sum_{U \in \mathcal{U}} q_U = 1$. Using similar arguments as in the proof of Proposition 7.4.3, we have the following inequalities:

$$\sum_{i \in V : d(i,j) \leq R} x_{ij}^U \leq q_U, \quad \forall j \in V, U \in \mathcal{U}$$

$$\sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \leq R} x_{ij}^U \geq p_j, \quad \forall j \in V$$

$$\sum_{i \in V} w_i y_i^U \leq q_U B, \quad \forall U \in \mathcal{U}$$

$$\sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x_{ij}^U \geq q_U t.$$

As mentioned before, if $|\text{RBall}(i, U_{\mathcal{S}}, R)| \geq \epsilon n$ then $i \notin \mathcal{S}$. Therefore,

$$y_i^U = \Pr[U_{\mathcal{S}} = U \wedge i \in \mathcal{S}] = 0, \quad \forall U \in \mathcal{U}, i \in V \setminus U, |\text{RBall}(i, U, R)| \geq \epsilon n.$$

The other constraints can be verified easily. We conclude that $(x, y, q) \in \mathcal{P}_{\mathsf{config2}}$ and $\mathcal{P}_{\mathsf{config2}} \neq \emptyset$. $\square$

Next, let us pick any $(x, y, q) \in \mathcal{P}_{\mathsf{config2}}$ and use the following algorithm to round it.

Algorithm 27 FRKnapCenterRound2 (x, y, q)

```
1: Randomly pick a set $U \in \mathcal{U}$ with probability $q_U$
2: Let $x'_{ij} \leftarrow x^U_{ij}/q_U$ and $y'_i \leftarrow \min\{y^U_i/q_U, 1\}$
3: $\mathcal{S}' \leftarrow$ BasicRFKnapCenterRound $(x', y')$
4: Let $i_1, i_2$ be vertices in $\mathcal{S}' \setminus U$ having largest weights.
5: return $\mathcal{S} = \mathcal{S}' \setminus \{i_1, i_2\}$
```

Analysis. Let us fix any $\gamma > 0$ and set $\epsilon := \frac{\gamma^2}{2}$. Also, let $\mathcal{E}(U)$ denote the event that $U \in \mathcal{U}$ is picked in the algorithm. Again, observe that (x', y') satisfies the following inequalities:

$$\sum_{j \in V} \sum_{i \in V: d(i,j) \le R} x'_{ij} \ge t,$$

$$\sum_{i \in V: d(i,j) \le R} x'_{ij} \le 1, \quad \forall j \in V,$$

$$\sum_{i \in V: d(i,j) \le R} x'_{ij} = \sum_{i \in V: d(i,j) \le R} x_{ij}/q_U, \quad \forall j \in V,$$

$$x'_{ij} \le y'_i, \quad \forall i, j \in V,$$

$$\sum_{i \in V} w_i y'_i \le B.$$

Recall that the algorithm BasicFRKnapCenterRound will return a solution $\mathcal{S}'$ consisting of a set $\mathcal{S}''$ with $w(\mathcal{S}'') \le B$ plus (at most) two extra "fractional" centers i^* and i^{**}. Moreover, we have $0 < y'_{i^*}, y'_{i^{**}} < 0$, which implies that $i^*, i^{**} \notin U$. Thus, by removing the two centers having highest weights in $\mathcal{S}' \setminus U$, we ensure that the total weight of $\mathcal{S}$ is within the given budget B with probability one.

Now we shall prove the coverage guarantee. By Proposition 7.4.2, $\mathcal{S}'$ covers at least t vertices within radius $3R$. If a vertex is blue, it can always be connected to some center in U; and hence, it is not affected by the removal of i_1, i_2. Because each of i_1 and i_2 can cover at most ϵn other red vertices, we have

$$|\mathcal{C}| \geq t - 2\epsilon n = 1 - \gamma^2 n.$$

For any $j \in V$, let X_j be the random indicator for the event that j is covered by $\mathcal{S}'$ (i.e., there is some $i \in \mathcal{S}'$ such that $d(i,j) \leq 3R$) but becomes unconnected due to the removal of i_1 or i_2. We say that j is a bad vertex iff $\mathbf{E}[X_j] \geq \gamma$. Otherwise, vertex j is said to be good. Note that $\sum_{j \in V} X_j \leq 2\epsilon n$ with probability one. Thus, there can be at most $2\epsilon n/\gamma$ bad vertices. Let T be the set of all good vertices. Then

$$|T| \geq n - 2\epsilon n/\gamma = (1-\gamma)n.$$

By Proposition 7.4.2, $\Pr[j \text{ is covered by } \mathcal{S}'] \geq p_j$. For any $j \in T$, we have

$$\begin{aligned}
\Pr[j \in \mathcal{C}] &= \Pr[j \text{ is covered by } \mathcal{S}' \wedge X_j = 0] \\
&= \Pr[j \text{ is covered by } \mathcal{S}'] - \Pr[j \text{ is covered by } \mathcal{S}' \wedge X_j = 1] \\
&\geq \Pr[j \text{ is covered by } \mathcal{S}'] - \Pr[X_j = 1] \\
&\geq p_j - \gamma.
\end{aligned}$$

This concludes the first part of Theorem 7.1.4 for the FRKnapCenter problem.

7.5 The Matroid Center problems with outliers

In this section, we will first give a tight 3-approximation algorithm for the RMatCenter problem, improving upon the 7-approximation algorithm by Chen et. al. [59]. Then we study the FRMatCenter problem and give a proof for the second part of Theorem 7.1.4.

7.5.1 The robust matroid center problem

Suppose $\mathcal{I} = (V, d, \mathcal{M}, t)$ is an instance the RMatCenter problem with the optimal radius R. Let $r_{\mathcal{M}}$ denote the rank function of $\mathcal{M}$. Consider the polytope $\mathcal{P}_{\mathsf{RMatCenter}}$ containing points (x, y) satisfying constraints (7.1)–(7.4), and the matroid rank constraints:

$$y(U) \leq r_{\mathcal{M}}(U), \quad \forall U \subseteq V. \tag{7.8}$$

Since R is the optimal radius, it is not difficult to check that $\mathcal{P}_{\mathsf{RMatCenter}} \neq \emptyset$. Let us pick any fractional solution $(x, y) \in \mathcal{P}_{\mathsf{RMatCenter}}$. The next step is to round (x, y) into an integral solution. Our 3-approximation algorithm is as follows.

Algorithm 28 RMatCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: Let $\mathcal{P}' := \left\{z \in [0,1]^V : z(U) \leq r_{\mathcal{M}}(U)\ \forall U \subseteq V\ \wedge\ z(F_i) \leq 1\ \ \forall i \in V'\right\}$
3: Find a basic solution $Y \in \mathcal{P}'$ which maximizes the linear function $f : [0,1]^V \to \mathbb{R}$
   defined as
       $$f(z) := \sum_{j \in V'} c_j \sum_{i \in F_j} z_i \quad \text{for } z \in [0,1]^V.$$
4: return $\mathcal{S} = \{i \in V : Y_i = 1\}$.
```

Analysis. Again, by construction, the clusters F_i are pairwise disjoint for $i \in V'$. Note that $\mathcal{P}'$ is the matroid intersection polytope between $\mathcal{M}$ and another partition matroid polytope saying that at most one item per set F_i for $i \in V'$ can be chosen. Moreover, $y \in \mathcal{P}'$ implies that $\mathcal{P}' \neq \emptyset$. Thus, $\mathcal{P}'$ has integral extreme points and optimizing over $\mathcal{P}'$ can be done in polynomial time. Note that the solution $\mathcal{S}$ is feasible as it satisfies the matroid constraint. The correctness of RMatCenterRound follows immediately by the following two propositions.

Proposition 7.5.1. *There are at least $f(Y)$ vertices in V that are at distance at most $3R$ from some open center in $\mathcal{S}$.*

Proof. Recall that $\mathcal{S}$ is the set of vertices $i \in V$ such that $Y_i = 1$. Moreover, by definition of $\mathcal{P}'$, there can be at most one open center in F_j (i.e., $|\mathcal{S} \cap F_j| \leq 1$) for each $j \in V'$ as $Y(F_j) \leq 1$. For any $j \in V'$,

- if $Y(F_j) = 0$, then there is no open center in F_j and its contribution in $f(Y)$ is zero,
- if $Y(F_j) = 1$, then we open some center $i \in F_j$ and the contribution of j to $f(Y)$ is equal to c_j. Recall that c_j is the number of clusters F_k such that $F_j \cap F_k \neq \emptyset$. By triangle inequality, the distance from k to i is at most $d(k,j) + d(j,i) \leq 2R + R = 3R$.

□

Proposition 7.5.2. *We have that $f(Y) \geq t$.*

Proof. For each $j \in V'$ and $i \in F_j$, define $y'_i := x_{ij}$ (this is well-defined as all clusters F_j for $j \in V'$ are pairwise disjoint). Also, set $y'_i := 0$ for other vertices i not belonging to any marked cluster. Then, by greedy choice and constraint (7.1), we have

$$f(y') = \sum_{j \in V'} c_j y'(F_j) = \sum_{j \in V'} c_j s_j \geq \sum_{j \in V} s_j \geq t.$$

By the choice of Y, we have $f(Y) \geq f(y') \geq t$. □

This analysis proves the second part of Theorem 7.1.1.

7.5.2 The fair robust matroid center problem

In this section, we consider the FRMatCenter problem. It is not difficult to modify and randomize algorithm RMCenterRound so that it would return a random solution satisfying both the fairness guarantee and matroid constraint, and preserving the coverage constraint *in expectation*. This can be done by randomly picking Y inside $\mathcal{P}'$. However, if we want to obtain some concrete guarantee on the coverage constraint, we may have to (slightly) violate either the matroid constraint or the fairness guarantee. We leave it as an open question whether there exists a true approximation algorithm for this problem.

We will start with a pseudo-approximation algorithm which always returns a basis of $\mathcal{M}$ plus at most one extra center. Our algorithm is quite involved. We first carefully round a fractional solution inside a matroid intersection polytope into a (random) point with a special property: the unrounded variables form a single path connecting some clusters and tight matroid rank constraints. Next, rounding this point will ensure that all but one cluster have an open center. Then opening one extra center is sufficient to cover at least t clients.

Finally, using a similar preprocessing step similar to the one in Section 7.4.2.3, we can correct the solution by removing the extra center without affecting the fairness and coverage guarantees by too much. This algorithm concludes Theorem 7.1.4.

7.5.2.1 A pseudo-approximation algorithm

Suppose $\mathcal{I} = (V, d, \mathcal{M}, t, \vec{p})$ is an instance the robust matroid center problem with the optimal radius R. Let $r_{\mathcal{M}}$ denote the rank function of $\mathcal{M}$ and $\mathcal{P}_{\mathcal{M}}$ be the matroid base polytope of $\mathcal{M}$. Consider the polytope $\mathcal{P}_{\mathsf{FRMatCenter}}$ containing points (x, y) satisfying constraints (7.1)–(7.4), the fairness constraint (7.5), and the matroid constraints (7.8). Using similar arguments as in the proof of Proposition 7.3.1, we can show that $\mathcal{P}_{\mathsf{FRMatCenter}}$ is a valid relaxation.

Proposition 7.5.3. *We have that* $\mathcal{P}_{\mathsf{FRMatCenter}} \neq \emptyset$.

Our algorithm will use the following rounding operation iteratively.

Algorithm 29 RoundSinglePoint $(y, \vec{r})$

```
1: $\delta^* \leftarrow \max\{\delta : z \in \mathcal{P}_{\mathcal{M}}; z_v = y_v + \delta r_v \ \forall v \in V\}$
2: $y' \leftarrow y + \delta^* \vec{r}$
3: **return** $(y', \delta^*)$
```

Given a point $y \in \mathcal{P}_{\mathcal{M}}$ and a vector $\vec{r}$, the procedure RoundSinglePoint will move y along direction $\vec{r}$ to a new point $y+\delta^*\vec{r}$ for some maximal $\delta^* > 0$ such that this point still lies in $\mathcal{P}_{\mathcal{M}}$. Note that one can find such a maximal δ^* in polynomial time. We will choose the initial point (x, y) as a vertex of $\mathcal{P}_{\mathsf{FRMatCenter}}$. By Cramer's rule, the entries of y will be rational with both numerators and denominators bounded by $O(2^n)$. The direction vector $\vec{r}$ also has this property by construction. Thus, it is not hard to verify that the maximal value of δ^* for which $y + \delta^* \vec{r} \in \mathcal{P}_{\mathcal{M}}$ is also rational and has both numerator and denominator at most $O(2^n)$ in every iteration. So we can compute δ^* exactly by a simple binary search.

The main algorithm is summarized in Algorithm 30, which can round any *vertex* point $(x, y) \in \mathcal{P}_{\mathsf{FRMatCenter}}$. Basically, we will round y iteratively. In each round, we construct a (multi)-bipartite graph where vertices on the left side are the disjoint sets $O_1, O_2, \ldots$ in Corollary 7.2.1. Vertices on the right side are corresponding to the disjoint sets $F_1, F_2, \ldots$ returned by RFiltering. Now each edge of the bipartite graph, connecting O_i and F_j, represents some unrounded variable $y_v \in (0, 1)$ where $v \in O_i$ and $v \in F_j$. See Figure 7.1.

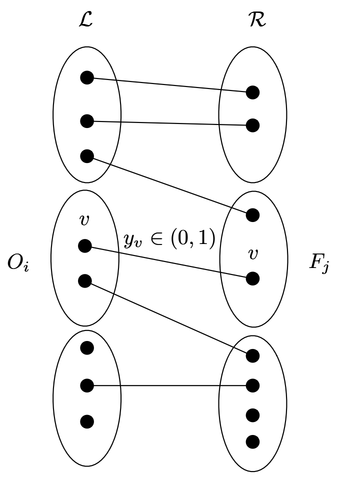

Figure 7.1: Construction of the multi-bipartite graph $H = (\mathcal{L}, \mathcal{R}, E_H)$ in the main algorithm.

Then we carefully pick a cycle (path) on this graph and round variables on the edges of this cycle (path). This is done by subroutines RoundCycle, RoundSinglePath, and RoundTwoPaths. See Figures 7.2, 7.3, and 7.4. Basically, these procedures will first choose a direction $\vec{r}$ which alternatively increases and decreases the variables on the cycle (path) so that (i) all tight matroid constraints are preserved and (ii) the number of (fractionally) covered clients is also preserved. Now we randomly move y along $\vec{r}$ or $-\vec{r}$ using procedure RoundSinglePoint to ensure that all the marginal probabilities are preserved.

Finally, all the remaining, fractional variables will form one path on the bipartite graph. We round these variables by the procedure RoundFinalPath which exploits the integrality of any face of a matroid intersection polytope. Then, to cover at least t clients, we may need to open one extra facility.

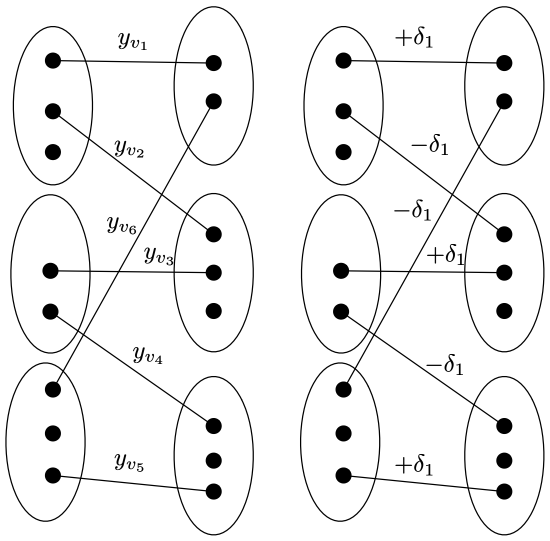

Figure 7.2: The left part shows a cycle. The right part shows how the variables on the cycle are being changed by RoundCycle.

Algorithm 30 PseudoFRMCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ and let $\mathcal{F} \leftarrow \{F_j : j \in V'\}$
2: Set $y'_i \leftarrow x_{ij}$ for all $j \in V', i \in F_j$
3: Set $y'_i \leftarrow 0$ for all $i \in V \setminus \bigcup_{j \in V'} F_j$
4: while $y'$ still contains some fractional values do
5:     Note that $y' \in \mathcal{P}_{\mathcal{M}}$. Compute the disjoint sets $O_1, \ldots, O_t$ and constants
       $b_{O_1}, \ldots, b_{O_t}$ as in Corollary 7.2.1.
6:     Let $O_0 \leftarrow V \setminus \bigcup_{i=1}^{t} O_i$ and $F_0 \leftarrow V \setminus \bigcup_{j \in V'} F_j$
7:     Construct a multi-bipartite graph $H = (\mathcal{L}, \mathcal{R}, E_H)$ where
         • each vertex $i \in \mathcal{L}$, where $\mathcal{L} = \{0, \ldots, t\}$, is corresponding to the set $O_i$
         • each vertex $j \in \mathcal{R}$, where $\mathcal{R} = \{0\} \cup \{k : F_k \in \mathcal{F}\}$, is corresponding to
           the set $F_j$
         • for each vertex $v \in V$ such that $y_v \in (0, 1)$: if $v$ belongs to some set $O_i$
           and $F_j$, add an edge $e$ with label $v$ connecting $i \in \mathcal{L}$ and $j \in \mathcal{R}$.
8:     Check the following cases (in order):
         • Case 1: $H$ contains a cycle. Let $\vec{v} = (v_1, v_2, \ldots, v_{2\ell})$ be the sequence of
           edge labels on this cycle. Update $y' \leftarrow$ RoundCycle$(y', \vec{v})$ and go to
           line 4.
         • Case 2: $H$ contains a maximal path with one endpoint in $\mathcal{L}$ and the
           other in $\mathcal{R}$. Let $\vec{v} = (v_1, v_2, \ldots, v_{2\ell+1})$ be the sequence of edge labels on
           this path. Update $y' \leftarrow$ RoundSinglePath$(y', \vec{v})$ and go to line 4.
         • Case 3: There are at least 2 distinct maximal paths (not necessarily dis-
           joint) having both endpoints in $\mathcal{R}$. Let $\vec{v}_1, \vec{v}_2$ be the sequences of edge la-
           bels on these two paths. Update $y' \leftarrow$ RoundTwoPaths$(y', \vec{v}_1, \vec{v}_2, \vec{c})$
           and go to line 4.
         • The remaining case: all edges in $H$ form a single path with both end-
           points in $\mathcal{R}$. Let $(v_1, v_2, \ldots, v_{2\ell})$ be the sequence of edge labels on this
           path. Let $Y \leftarrow$ RoundFinalPath$(y', \vec{v})$ and exit the loop.
9: return $\mathcal{S} = \{i \in V : Y_i = 1\}$.
```

Algorithm 31 RoundCycle $(y', \vec{v})$

```
1: Initialize $\vec{r} = \vec{0}$, then set $r_{v_j} = (-1)^j$ for $j = 1, 2, \ldots, |\vec{v}|$
2: $(y_1, \delta_1) \leftarrow$RoundSinglePoint$(y', \vec{r})$
3: return $y_1$
```

Algorithm 32 RoundSinglePath $(y', \vec{v})$

```
1: Initialize $\vec{r} = \vec{0}$, then set $r_{v_j} = (-1)^{j+1}$ for $j = 1, 2, \ldots, |\vec{v}|$
2: $(y_1, \delta_1) \leftarrow$RoundSinglePoint$(y', \vec{r})$
3: return $y_1$
```

Algorithm 33 RoundTwoPaths $(y', \vec{v}, \vec{v}', \vec{c})$

```
1: WLOG, suppose $j_1, j_2 \in \mathcal{R}$ are endpoints of $v_1, v_{2\ell}$ of the path $\vec{v}$ respectively
   and $c_{j_1} \geq c_{j_2}$
2: WLOG, suppose $j'_1, j'_2 \in \mathcal{R}$ are endpoints of $v'_1, v'_{2\ell'}$ of the path $\vec{v}'$ respectively
   and $c_{j'_1} \geq c_{j'_2}$
3: $\Delta_1 \leftarrow c_{j_1} - c_{j_2}$;    $\Delta_2 \leftarrow c_{j'_1} - c_{j'_2}$;    $\vec{r} \leftarrow \vec{0}$
4: $V_1^+ \leftarrow \{v_1, v_3, \ldots, v_{2\ell-1}\}; V_1^- \leftarrow \{v_2, v_4, \ldots, v_{2\ell}\}$
5: $V_2^+ \leftarrow \{v'_2, v'_4, \ldots, v'_{2\ell'}\}; V_2^- \leftarrow \{v'_1, v'_3, \ldots, v'_{2\ell'-1}\}$
6: for each $v \in V_1^+$: $r_v \leftarrow r_v + 1$; for each $v \in V_1^-$: $r_v \leftarrow r_v - 1$
7: for each $v \in V_2^+$: $r_v \leftarrow r_v + \Delta_1/\Delta_2$; for each $v \in V_2^-$: $r_v \leftarrow r_v - \Delta_1/\Delta_2$
8: $(y_1, \delta_1) \leftarrow$RoundSinglePoint$(y', \vec{r})$
9: $(y_2, \delta_2) \leftarrow$RoundSinglePoint$(y', -\vec{r})$
10: With probability $\delta_1/(\delta_1 + \delta_2)$: return $y_2$
11: With remaining probability $\delta_2/(\delta_1 + \delta_2)$: return $y_1$
```

Algorithm 34 RoundFinalPath $(y, \vec{v})$

```
1: $\mathcal{P}_1 \leftarrow \{z \in [0,1]^V : z(U) \leq r_{\mathcal{M}}(U)\ \forall U \subseteq V \wedge z(O_i) = b_{O_i}\ \forall i \in \mathcal{L} \setminus \{0\} \wedge z_i = 0\ \forall i : y_i = 0\}$
2: $\mathcal{P}_2 \leftarrow \{z \in [0,1]^V : z(F_j) = y(F_j)\ \forall j \in V' \setminus J \wedge z(F_j) \leq 1\ \forall j \in J\}$, where $J \subseteq \mathcal{R}$
   is the set of vertices in $\mathcal{R}$ on the path $\vec{v}$.
3: Pick an arbitrary extreme point $\hat{y}$ of $\mathcal{P}' = \mathcal{P}_1 \cap \mathcal{P}_2$
4: for each $j \in \mathcal{R}$ and $j$ is on the path $\vec{v}$: if $\hat{y}(F_j) = 0$, pick an arbitrary $u \in F_j$
   and set $\hat{y}_u \leftarrow 1$.
5: return $\hat{y}$
```

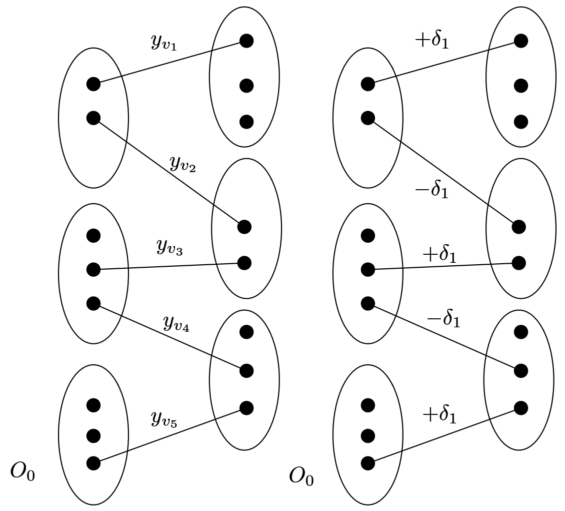

Figure 7.3: The left part shows a single path. The right part shows how the variables on the path are being changed by RoundSinglePath.

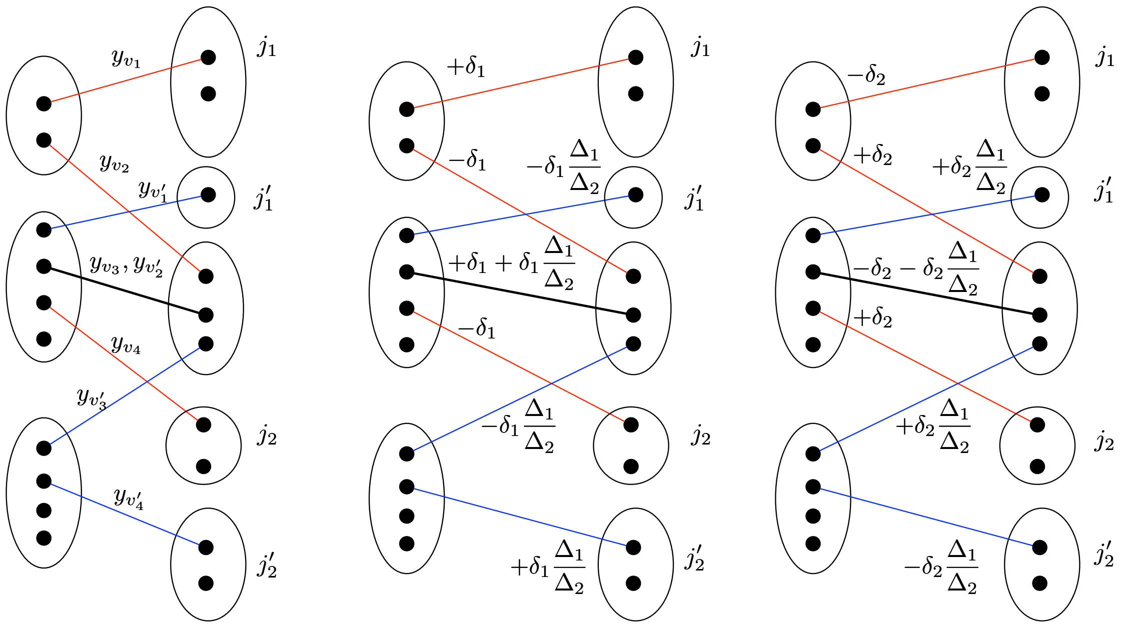
y_{v_1}
j_1
y_{v_2}
$y_{v'_1}$
j'_1
$y_{v_3}, y_{v'_2}$
y_{v_4}
$y_{v'_3}$
j_2
$y_{v'_4}$
j'_2
$+\delta_1$
$-\delta_1$
$-\delta_1 \frac{\Delta_1}{\Delta_2}$
$+\delta_1 + \delta_1 \frac{\Delta_1}{\Delta_2}$
$-\delta_1$
$-\delta_1 \frac{\Delta_1}{\Delta_2}$
$+\delta_1 \frac{\Delta_1}{\Delta_2}$
$-\delta_2$
$+\delta_2$
$+\delta_2 \frac{\Delta_1}{\Delta_2}$
$-\delta_2 - \delta_2 \frac{\Delta_1}{\Delta_2}$
$+\delta_2$
$+\delta_2 \frac{\Delta_1}{\Delta_2}$
$-\delta_2 \frac{\Delta_1}{\Delta_2}$

Figure 7.4: The left part shows an example of two distinct maximal paths chosen in Case 3. The black edge is common in both paths. The middle and right parts are two possibilities of rounding y. With probability $\delta_1/(\delta_1 + \delta_2)$, the strategy in the right part is adopted. Otherwise, the strategy in the middle part is chosen.

7.5.2.2 Analysis of PseudoFRMCenterRound

Proposition 7.5.4. *In all but the last iteration, the while-loop (lines* 4 *to* 8*) of* PseudoFRMCenterRound *preserves the following invariant: if y' lies in the face D of $\mathcal{P}_{\mathcal{M}}$ (w.r.t all tight matroid rank constraints) at the beginning of the iteration, then $y' \in D$ at the end of this iteration.*

Proof. Observe that $y' \in \mathcal{P}_{\mathcal{M}}$ at the beginning of the first iteration due to the definition of y'. Fix any iteration. Let y'' be the updated y' at the end of the iteration. Now, by Corollary 7.2.1, it suffices to show that

$$y'' \in \{x \in \mathbb{R}^n : x(S) = b_S \ \ \forall S \in \emptyset; \quad x_i = 0 \ \ \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}}\},$$

where $J \subseteq V$ is the set of all vertices i that $y'_i = 0$. Note that y'' is the output of one of the three subroutines RoundCycle, RoundSinglePath, and RoundTwoPaths. Since we only round floating variables strictly greater than zero, we have that $y''_i = 0$ for all $i \in J$. Also, the procedure RoundSinglePoint guarantees that $y'' \in \mathcal{P}_{\mathcal{M}}$.

- When calling the procedure RoundCycle, observe that each vertex $j \in \mathcal{L}$ on the cycle is adjacent to exactly two edges. By construction, we always increase the variable on one edge and decrease the variable on the other edge at the same rate. See Figure 7.2. Therefore, $y''(O_j) = y'(O_j) = b_{O_j}$ for all $j \in \emptyset$.
- When calling the procedure RoundSinglePath, recall that our path is maximal and has one endpoint in $\mathcal{L}$ and the other in $\mathcal{R}$. We claim that the left end-

point of this path should be corresponding to the set O_0. Otherwise, suppose it is some set O_j with $j > 0$. We have the tight constraint $y'(O_j) = b_{O_j} \in \mathbb{Z}^+$. Then the degree of the vertex j must be at least 2 as there must be at least two fractional variables in this set. This contradicts to the fact that our path is maximal. See Figure 7.3. By the same argument as before, we have that $y''(O_j) = y'(O_j) = b_{O_j}$ for all $j \in \emptyset$.

- In the procedure RoundTwoPaths, we round the variables on two paths which have both endpoints in $\mathcal{R}$. Thus, any vertex j should be adjacent to either 2 or 4 edges. Again, by construction, the net change in $y'(O_j)$ is equal to zero. See Figure 7.4.

Finally, the claim follows by induction. □

Proposition 7.5.5. PseudoFRMCenterRound *terminates in polynomial time.*

Proof. Note that, in each iteration, each floating variable $y'_v \in (0,1)$ is corresponding to exactly one edge in the bipartite graph. This is because, by construction, the sets $O_0, \ldots, O_t$ form a partition of V and the sets in $\mathcal{F}$ and F_0 also form a partition of V. Thus, as long as there are fractional values in y', our graph will have some cycle or path.

Now we will show that the while-loop (lines 4 to 8) terminates after $O(|V|)$ iterations. For any set S, let $\chi(S)$ denote the characteristic vector of S. That is, $\chi(v) = 1$ for $v \in S$ and $\chi(v) = 0$ otherwise. Let us fix any iteration and let $\mathcal{T} = \{\chi(S) : S \subseteq V \ \wedge \ y'(S) = r_{\mathcal{M}}(S)\}$ be the set of all tight constraints. In this iteration, we will move y' along some direction $\vec{r}$ as far as possible (by procedure

RoundSinglePoint). It means that the new point $y'' = y' + \delta^* \vec{r}$ will either have at least one more rounded variable or hit a new tight constraint $y''(S_0) = r_{\mathcal{M}}(S_0)$ (while $y'(S_0) < r_{\mathcal{M}}(S_0)$) for some $S_0 \subseteq V$. Indeed $\chi(S_0)$ is linearly independent of all vectors in $\mathcal{T}$.

Proposition 7.5.4 says that all the tight constraints are preserved in the rounding process. Therefore, in the next iteration, we either have at least one more rounded variable or the rank of $\mathcal{T}$ is increased by at least 1. This implies the algorithm terminates after at most $|V|$ iterations. □

Proposition 7.5.6. *In all iterations, the while-loop (lines* 4 *to* 8*) of* PseudoFRMCenterRound *satisfies the invariant that* $y'(F_j) \leq 1$ *for all* $F_j \in \mathcal{F}$*.*

Proof. By constraints 7.2 and 7.3, this property is true at the beginning of the first iteration. By a very similar argument as in the proof of Proposition 7.5.4, this is also true during all but the last iteration. (Note that if j is an endpoint of a path, then j must be adjacent to exactly one fractional value y'_v, which could be rounded to one, while other variables $\{y'_{v'} : v' \in F_j, v' \neq v\}$ are already rounded to zero as our path is maximal.) Finally, it is not hard to check that procedure RoundFinalPath also does not violate this invariant. □

Proposition 7.5.7. PseudoFRMCenterRound *returns a solution* $\mathcal{S}$ *which is some independent set of* $\mathcal{M}$ *plus (at most) one extra vertex in* V*.*

Proof. Let us focus on the procedure RoundFinalPath. Recall that the polytope

$\mathcal{P}'$ in RoundFinalPath is the intersection of the following two polytopes:

$$\mathcal{P}_1 = \left\{ z \in [0,1]^V : z(U) \leq r_{\mathcal{M}}(U) \ \forall U \subseteq V \wedge z(O_i) = b_{O_i} \ \forall i \in \mathcal{L} \setminus \{0\} \wedge z_i = 0 \ \forall i : y_i = 0 \right\},$$

and

$$\mathcal{P}_2 = \{ z \in [0,1]^V : z(F_j) = y(F_j) \ \forall j \in V' \setminus J \wedge z(F_j) \leq 1 \ \forall j \in J \},$$

where $J \subseteq \mathcal{R}$ is the set of vertices in $\mathcal{R}$ on the path $\vec{v}$.

By construction, $\mathcal{P}_1$ is the face of the matroid base polytope $\mathcal{P}_{\mathcal{M}}$ corresponding to all tight constraints of y. It is well-known that $\mathcal{P}_1$ itself is also a matroid base polytope. By Propositions 7.5.4 and 7.5.6, we have that $y \in \mathcal{P}_1$ and $y \in \mathcal{P}_2$. Thus, $y \in \mathcal{P}$ which implies that $\mathcal{P} \neq \emptyset$. Moreover, $\mathcal{P}_2$ is a partition matroid polytope. (Observe that $z(F_j) = y(F_j) \in \{0,1\} \ \forall j \in V' \setminus J$ since all fractional variables are on the path $\vec{v}$.) Therefore, $\mathcal{P} = \mathcal{P}_1 \cap \mathcal{P}_2$ has integral extreme points and the point $\hat{y}$ chosen in line 3 is integral.

Finally, recall that $\vec{v} = (v_1, v_2, \ldots, v_{2\ell})$ is a simple path with both endpoints in $\mathcal{R}$. The constraints of $\mathcal{P}_1$ and integrality of b_{O_i}'s ensure that $\hat{y}_{v_1} + \hat{y}_{v_2} = 1, \hat{y}_{v_3} + \hat{y}_{v_4} = 1, \ldots, \hat{y}_{v_{2\ell-1}} + \hat{y}_{v_{2\ell}} = 1$. In other words, every vertex $i \in \mathcal{L}$ on the path will be "matched" with exactly one vertex in $\mathcal{R}$. Thus, there can be at most one vertex $j \in \mathcal{R}$ on the path such that $\hat{y}(F_j) = 0$ in line 4. Opening $u \in F_j$ adds one extra facility to our solution. □

Recall that $\mathcal{C}$ is the (random) set of all clients within radius $3R$ from some center in $\mathcal{S}$, where R is the optimal radius. The following two propositions will conclude our analysis.

Proposition 7.5.8. *$|\mathcal{C}| \geq t$ with probability one.*

Proof. Let f denote the function defined in Algorithm RMCenterRound (i.e., $f(z) = \sum_{j \in V'} \sum_{i \in F_i} z_i$ for any $z \in [0,1]^V$.) Using a similar argument as in the proof of Proposition 7.5.1, one can easily verify that there are at least $f(Y)$ vertices in V that are within radius $3R$ from some open center in $\mathcal{S}$. Next, it suffices to show that $f(Y) \geq t$.

By definition of y' in lines 2 and 3, we have that $f(y') \geq t$ (see the proof of Proposition 7.5.2.) We now claim that $f(y')$ is not decreasing after each iteration of the rounding scheme. We check the following cases:

- Case y' is rounded by RoundCycle: observe that $y'(F_j)$ is preserved for all $j \in \mathcal{R}$ since j is adjacent to two edges and we increase/decrease the corresponding variables by the same amount. Thus, $f(y')$ is unchanged.

- Case y' is rounded by RoundSinglePath: if $j \in \mathcal{R}$ is not the endpoint of the path then j is adjacent to two edges on the path and $y'(F_j)$ is unchanged. If j is the endpoint, then we increase the variable on the adjacent edge; and hence, $y'(F_j)$ will increase. See Figure 7.3.

- Case y' is rounded by RoundTwoPaths: again, for any $j \in \mathcal{R} \backslash \{j_1, j_2, j'_1, j'_2\}$, we have that $y'(F_j)$ remains unchanged in the process. We now verify the change in f caused by the four endpoints j_1, j_2, j'_1, and j'_2. Suppose y_1 is returned, the contribution of these points in $f(y_1)$ is

$$\begin{aligned}
&c_{j_1}y_1(F_{j_1}) + c_{j_2}y_1(F_{j_2}) + c_{j_1'}y_1(F_{j_1'}) + c_{j_2'}y_1(F_{j_2'}) \\
&= c_{j_1}(y'(F_{j_1}) + \delta_1) + c_{j_2}(y'(F_{j_2}) - \delta_1) + c_{j_1'}\left(y(F_{j_1'}) - \delta_1\frac{\Delta_1}{\Delta_2}\right) + c_{j_2'}\left(y(F_{j_2'}) + \delta_1\frac{\Delta_1}{\Delta_2}\right) \\
&= c_{j_1}y'(F_{j_1}) + c_{j_2}y'(F_{j_2}) + c_{j_1'}y'(F_{j_1'}) + c_{j_2'}y'(F_{j_2'}) + \delta_1(c_{j_1} - c_{j_2}) + \delta_1\frac{\Delta_1}{\Delta_2}(c_{j_2'} - c_{j_1'}) \\
&= c_{j_1}y'(F_{j_1}) + c_{j_2}y'(F_{j_2}) + c_{j_1'}y'(F_{j_1'}) + c_{j_2'}y'(F_{j_2'}).
\end{aligned}$$

Hence, $f(y_1) = f(y')$. Similarly, one can verify that $f(y_2) = f(y')$.

- Case y' is rounded by RoundFinalPath: we have shown in the proof of Proposition 7.5.7 that $y'(F_j) = 1$ for all $j \in J$ where J is the set of vertices in $\mathcal{R}$ on the path $\vec{v}$. This fact and the other constraints of $\mathcal{P}_2$ ensure that $y'(F_j)$ is not decreasing for all $j \in V'$.

□

Proposition 7.5.9. $\Pr[j \in \mathcal{C}] \geq p_j$ *for all* $j \in V$.

Proof. Let y' be the vector defined as in lines 2 and 3 of PseudoFRMCenterRound. It suffices to show that, for all $j \in V'$, $\Pr[Y(F_j) = 1] \geq y'(F_j)$. (Note that $y'(F_j) \geq p_j$ by constraint (7.5).) This is because, for any vertex $k \in V \setminus V'$, the algorithm RFiltering guarantees that there exists $j \in V'$ such that $F_k \cap F_j \neq \emptyset$, and $y'(F_j) = \sum_{i \in V : d(i,j) \leq R} x_{ij} \geq \sum_{i \in V : d(i,k) \leq R} x_{ik} = y'(F_k)$. Notice that the event $Y(F_j) = 1$ means there is some open center F_j and the distance from k to this center should be at most $3R$. Thus,

$$\Pr[k \in \mathcal{C}] \geq \Pr[Y(F_j) = 1] \geq y'(F_j) \geq y'(F_k) \geq p_k,$$

by constraint (7.5).

Fix any $j \in V'$. Recall that Y is obtained by rounding y' and, by Proposition 7.5.6 and the proof of 7.5.7, we have $Y(F_j) \in \{0, 1\}$ and $\Pr[Y(F_j) = 1] = \mathbf{E}[Y(F_j)]$. We now show that the expected value of $y'(F_j)$ does not decrease after each iteration of the while-loop.

- Case y' is rounded by RoundCycle: $y'(F_j)$ is unchanged as before.

- Case y' is rounded by RoundSinglePath: if j is not the endpoint of $\vec{v}$ then $y'(F_j)$ is unchanged. Otherwise, $y'(F_j)$ is increase by some $\delta_1 > 0$ with probability one.

- Case y' is rounded by RoundTwoPaths: again, if $j \notin \{j_1, j_2, j'_1, j'_2\}$ then $y'(F_j)$ is unchanged. Now suppose $j = j_1$. With probability $\delta_1/(\delta_1 + \delta_2)$, $y'(F_{j_1})$ is increase by δ_2, and, with the remaining probability, it is decreased by δ_1. Thus, the expected change in $y'(F_{j_1})$ is

$$\frac{\delta_1}{\delta_1 + \delta_2}(\delta_2) + \frac{\delta_2}{\delta_1 + \delta_2}(-\delta_1) = 0.$$

  Similarly, one can verify that the expected values of $y'(F_{j_2}), y'(F_{j'_1})$, and $y'(F_{j'_2})$ remain the same.

- Case y' is rounded by RoundFinalPath: we have showed in the proof of Proposition 7.5.7 that if j is on the path $\vec{v}$, then $Y(F_j) = 1$. Otherwise, the constraints of $\mathcal{P}_2$ ensure that $Y(F_j) = y'(F_j)$.

$\square$

So far we have proved the following theorem.

Theorem 7.5.1. PseudoFRMCenterRound *will return a random solution $\mathcal{S}$ such that*

- *$\mathcal{S}$ is the union of some basis of $\mathcal{M}$ with (at most) one extra vertex,*
- *$|\mathcal{C}| \geq t$ with probability one,*
- *$\Pr[j \in \mathcal{C}] \geq p_j$ for all $j \in V$.*

7.5.2.3 An algorithm satisfying the matroid constraint exactly

Using a similar technique as in Section 7.4.2.3, we will develop an approximation algorithm for the FRMatCenter problem which always returns a feasible solution. Let $\epsilon > 0$ a small parameter to be determined. Let $\mathcal{U}$ denote the collection of all possible sets of verticies with size at most $\lceil 1/\epsilon \rceil$ such that U is an independent set of $\mathcal{M}$. Again, we have that $|\mathcal{U}| \leq n^{O(1/\epsilon)}$. Suppose R is the optimal radius to our instance. For any $i \in V$, recall that $\mathrm{RBall}(i, U, R)$ is the set of red vertices within radius $3R$ from i.

Consider the *configuration* polytope $\mathcal{P}_{\mathsf{config3}}$ containing points (x, y, q) with the following constraints:

$$\begin{cases} \sum_{U\in\mathcal{U}} q_U = 1 & \\ \sum_{i\in V: d(i,j)\le R} x_{ij}^U \le q_U & \forall j \in V, U \in \mathcal{U} \\ \sum_{U\in\mathcal{U}} \sum_{i\in V: d(i,j)\le R} x_{ij}^U \ge p_j & \forall j \in V \\ x_{ij}^U \le y_i^U & \forall i, j \in V, U \in \mathcal{U} \\ \sum_{i\in W} y_i^U \le q_U r_{\mathcal{M}}(W) & \forall U \in \mathcal{U}, W \subseteq V \\ \sum_{j\in V} \sum_{i\in V: d(i,j)\le R} x_{ij}^U \ge q_U t & \\ y_i^U = 1 & \forall U \in \mathcal{U}, i \in U \\ y_i^U = 0 & \forall U \in \mathcal{U}, i \in V\setminus U, |\mathrm{RBall}(i, U, R)| \ge \epsilon n \\ x_{ij}^U, y_i^U, q_U \ge 0 & \forall i, j \in V, U \in \mathcal{U} \end{cases}$$

We first claim that $\mathcal{P}_{\mathsf{config3}}$ is a valid relaxation polytope for the problem.

Proposition 7.5.10. *We have that $\mathcal{P}_{\mathsf{config3}} \neq \emptyset$.*

Proof. Suppose $\mathcal{S}$ is a solution drawn from the optimal distribution $\mathcal{D}$. We compute a subset $U_{\mathcal{S}}$ of $\mathcal{S}$ using a similar procedure as in the proof of Proposition 7.4.4. Recall that $|\mathrm{RBall}(i, U_{\mathcal{S}}, R)| < \epsilon n$ for all $i \in \mathcal{S}\setminus U_{\mathcal{S}}$ and $|U_{\mathcal{S}}| \le \lceil 1/\epsilon \rceil$. Since $U_{\mathcal{S}} \subseteq \mathcal{S}$, $U_{\mathcal{S}}$ is also an independent set of $\mathcal{M}$, implying that $U_{\mathcal{S}} \in \mathcal{U}$.

Now for any $U \in \mathcal{U}$, we set $q_U := \Pr[U_{\mathcal{S}} = U]$. Let x_{ij}^U be probability of the joint event: $U_{\mathcal{S}} = U$ and j is connected to i. Finally, let y_i^U be the probability of the joint event: $U_{\mathcal{S}} = U$ and $i \in \mathcal{S}$. Then it is clear that $\sum_{U \in \mathcal{U}} q_U = 1$. Using similar arguments as in the proofs of Propositions 7.4.4 and 7.4.3, we have the following inequalities:

$$\sum_{i \in V : d(i,j) \leq R} x_{ij}^U \leq q_U, \quad \forall j \in V, U \in \mathcal{U} \tag{7.9}$$

$$\sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \leq R} x_{ij}^U \geq p_j, \quad \forall j \in V \tag{7.10}$$

$$\sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x_{ij}^U \geq q_U t, \tag{7.11}$$

$$y_i^U = 0 \quad \forall U \in \mathcal{U}, i \in V \setminus U, |\mathrm{RBall}(i, U, R)| \geq \epsilon n. \tag{7.12}$$

Recall that y_i^U / q_U is the probability that $i \in \mathcal{S}$ conditioned on $U = U_{\mathcal{S}}$. Since $\mathcal{S}$ is independent with probability one, we have $|\mathcal{S} \cap W| \leq r_{\mathcal{M}}(W)$ for all $W \subseteq V$. Therefore,

$$\begin{aligned} r_{\mathcal{M}}(W) &\geq \mathbf{E}[|\mathcal{S} \cap W| \mid U = U_{\mathcal{S}}] \\ &= \sum_{i \in W} \Pr[i \in \mathcal{S} \mid U = U_{\mathcal{S}}] \\ &= \sum_{i \in W} y_i^U / q_U, \end{aligned}$$

for all $W \subseteq V$.

The other constraints can be verified easily. We conclude that $(x, y, q) \in \mathcal{P}_{\mathsf{config3}}$ and $\mathcal{P}_{\mathsf{config3}} \neq \emptyset$. □

Next, let us pick any $(x, y, q) \in \mathcal{P}_{\mathsf{config3}}$ and use the following algorithm to round it.

Algorithm 35 FRMCenterRound (x, y, q)

```
1: Randomly pick a set $U \in \mathcal{U}$ with probability $q_U$
2: Let $x'_{ij} \leftarrow x^U_{ij}/q_U$ and $y'_i \leftarrow \min\{y^U_i/q_U, 1\}$
3: $\mathcal{S}' \leftarrow$ PseudoFRMCenterRound $(x', y')$
4: Let $i^*$ be the "extra" vertex in $\mathcal{S}'$.
5: return $\mathcal{S} = \mathcal{S}' \setminus \{i\}$
```

Analysis. We are now ready to prove the second part of Theorem 7.1.4. Let us fix any $\gamma > 0$ and set $\epsilon := \gamma^2$. Also, let $\mathcal{E}(U)$ denote the event that $U \in \mathcal{U}$ is picked in the algorithm. Note that (x', y') satisfies the following inequalities:

$$\sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x'_{ij} \geq t,$$

$$\sum_{i \in V : d(i,j) \leq R} x'_{ij} \leq 1, \quad \forall j \in V,$$

$$\sum_{i \in V : d(i,j) \leq R} x'_{ij} = \sum_{i \in V : d(i,j) \leq R} x_{ij}/q_U, \quad \forall j \in V,$$

$$x'_{ij} \leq y'_i, \quad \forall i, j \in V,$$

$$\sum_{i \in W} y'_i \leq r_{\mathcal{M}}(W), \qquad \forall W \subseteq V.$$

Moreover, $y'_i = 1$ for all $i \in U$ and $y'_i = 0$ for all $i \in V \setminus U$ and $\mathrm{RBall}(i, U, R) \geq \epsilon n$.

Recall that the algorithm PseudoFRMCenterRound will return a solution $\mathcal{S}'$ is the union of a basis of $\mathcal{M}$ with an extra center i^*. Moreover, we have $0 < y'_{i^*} < 0$, which implies that $i^* \notin U$. Thus, by removing i^* from $\mathcal{S}'$, we ensure that the resulting set is a basis of $\mathcal{M}$ with probability one.

Now we shall prove the coverage guarantee. By Theorem 7.5.1, $\mathcal{S}'$ covers at least t vertices within radius $3R$. If a vertex is blue, it can always be connected to some center in U; and hence, it is not affected by the removal of i_1, i_2. Because each of i^* can cover at most ϵn other red vertices, we have

$$|\mathcal{C}| \geq t - \epsilon n = 1 - \gamma^2 n.$$

For any $j \in V$, let X_j be the random indicator for the event that j is covered by $\mathcal{S}'$ (i.e., there is some $i \in \mathcal{S}'$ such that $d(i, j) \leq 3R$) but becomes unconnected due to the removal of i^*. We say that j is a bad vertex iff $\mathbf{E}[X_j] \geq \gamma$. Otherwise, vertex j is said to be good. Again, $\sum_{j \in V} X_j \leq \epsilon n$ with probability one. Thus, there can be at most $\epsilon n/\gamma$ bad vertices. Let T be the set of all good vertices. Then

$$|T| \geq n - \epsilon n/\gamma = (1 - \gamma)n.$$

By Theorem 7.5.1, $\Pr[j \text{ is covered by } \mathcal{S}'] \geq p_j$. So, for any $j \in T$, we have

$$\Pr[j \in \mathcal{C}] \geq \Pr[j \text{ is covered by } \mathcal{S}'] - \Pr[X_j = 1] \geq p_j - \gamma.$$

Chapter 8: Future Work

In this thesis, we provide new dependent rounding techniques and approximation algorithms for several FL problems. There are still wide open problems in this area for further research. First of all, can we make the rounding schemes in Chapter 2 work for other constraints such as multiple linear constraints, a single matroid constraint, or a matroid intersection constraint? One major direction is to study the relationship between maximum-entropy distributions and the near-independence property. (We have already known that such distributions on bases of uniform matroids or spanning trees provide negative correlation.) It is well-known that rounding point inside a matroid intersection polytope may result in significant positive correlation between the variables. One interesting question is whether we can sacrifice some marginal preservation to gain (partial) negative correlation? Also, can our techniques be applied to capacitated versions of FL problems?

Secondly, the gap between the best-known lower-bound ($1 + 2/e \approx 1.735$) and the current approximation ratios for both k-median (2.675) and KM (17.46) problems remains quite large. Can we obtain improved approximation algorithms for these problems? The technique in Chapter 4 could be useful for KM: it allows us to open any extra constant number of facilities without too much loss in the total connection cost. Furthermore, the ideas of neighborhood tree and randomized rounding in Chapter 4 also lead to a simple 8-approximation algorithm for the Matroid Median problem (the problem is already known to be 8-approximable [12] though the analysis here is simpler.) The main difficulty is that we need to round a vector subject to a matroid intersection constraint, which may require a novel dependent rounding technique. (Again, the technique by Chekuri et. al. [82] does not seem to work here as it does not always return a basis.)

We provided randomized algorithms for the classic k-center and knapsack center problems which guarantee both a worst-case bound and a much better average bound in Chapter 5 and Chapter 6. For the k-center problem, the current worst-case ratio is 3. So one natural question is whether there exists a randomized algorithm which not only matches the tight worst-case ratio of 2 but also gives a less-than-2 average ratio. Furthermore, can we generalize these results to other variants such as fault-tolerant or capacitated version?

Lastly, there are a few open questions regarding the lottery model in Chapter 7. We leave it as an open question if there are approximation algorithms for the FRkCenter, FRKnapCenter, and FRMatCenter problems which do not violate the coverage and fairness constraints. Analyzing the expected distance of connected clients in these problems is also interesting. Can we obtain some non-trivial bounds in this case? In addition, it is natural to apply the lottery model to k-center and knapsack center problems: here we look for a distribution $\mathcal{D}$ such that the maximum expected connection cost of any client is minimized. Using known results for the k-median problem [33], it is not difficult to obtain a 3.25-approximation for this problem. We may need better techniques to beat this approximation ratio.

Bibliography

[1] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[2] P. Crescenzi and V. Kann. *Approximation on the web: A compendium of NP optimization problems.* Springer Berlin Heidelberg, Berlin, Heidelberg, 1997.

[3] Giorgio Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1999.

[4] Oscar H. Ibarra and Chul E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *J. ACM*, 22(4):463–468, October 1975.

[5] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*, STOC '06, pages 681–690, New York, NY, USA, 2006. ACM.

[6] Ivan D. Baev and Rajmohan Rajaraman. Approximation algorithms for data placement in arbitrary networks. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA.*, pages 661–670, 2001.

[7] Ivan D. Baev, Rajmohan Rajaraman, and Chaitanya Swamy. Approximation algorithms for data placement problems. *SIAM J. Comput.*, 38(4):1411–1429, 2008.

[8] Zachary Friggstad and Mohammad R. Salavatipour. Minimizing movement in mobile facility location problems. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 357–366, 2008.

[9] Sara Ahmadian, Zachary Friggstad, and Chaitanya Swamy. Local-search based approximation algorithms for mobile facility location problems. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1607–1621, 2013.

[10] Inge Li Gørtz and Viswanath Nagarajan. Locating depots for capacitated vehicle routing. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, pages 230–241, 2011.

[11] Deeparnab Chakrabarty and Chaitanya Swamy. Facility location with client latencies: Linear programming based techniques for minimum latency problems. In *Integer Programming and Combinatoral Optimization - 15th International Conference, IPCO 2011, New York, NY, USA, June 15-17, 2011. Proceedings*, pages 92–103, 2011.

[12] Chaitanya Swamy. Improved approximation algorithms for matroid and knapsack median problems and applications. In *APPROX/RANDOM 2014*, volume 28, pages 403–418, 2014.

[13] Mark S. Daskin, Lawrence V. Snyder, and Rosemary T. Berger. *Logistics Systems: Design and Optimization*, chapter Facility Location in Supply Chain Design, pages 39–65. Springer US, Boston, MA, 2005.

[14] Alan S. Manne. Plant location under economies-of-scale—decentralization and computation. *Manage. Sci.*, 11(2):213–235, November 1964.

[15] Delbert Dueck, Brendan J. Frey, Nebojsa Jojic, Vladimir Jojic, Guri Giaever, Andrew Emili, Gabe Musso, and Robert Hegele. Constructing treatment portfolios using affinity propagation. *Research in Computational Molecular Biology: 12th Annual International Conference, RECOMB 2008, Singapore, March 30 - April 2, 2008. Proceedings*, pages 360–371, 2008.

[16] H. Li. Two-view motion segmentation from linear programming relaxation. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[17] N. Lazic, I. Givoni, B. Frey, and P. Aarabi. Floss: Facility location for subspace segmentation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 825–832, Sept 2009.

[18] Denis Krivitski, Assaf Schuster, and Ran Wolff. A local facility location algorithm for large-scale distributed systems. *Journal of Grid Computing*, 5(4):361–378, 2007.

[19] Sudipto Guha, Adam Meyerson, and Kamesh Munagala. Hierarchical placement and network design problems. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 603–612, 2000.

[20] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 291–300, 2004.

[21] Moses Charikar and Rina Panigrahy. Clustering to minimize the sum of cluster diameters. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 1–10, New York, NY, USA, 2001. ACM.

[22] Gagan Aggarwal, Rina Panigrahy, Tomás Feder, Dilys Thomas, Krishnaram Kenthapadi, Samir Khuller, and An Zhu. Achieving anonymity via clustering. *ACM Transactions on Algorithms (TALG)*, 6(3):49:1–49:19, July 2010.

[23] Dorit S. Hochbaum. Heuristics for the fixed cost median problem. *Mathematical Programming*, 22(1):148–162.

[24] David B. Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 265–274, New York, NY, USA, 1997. ACM.

[25] Kamal Jain and Vijay V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *Journal of the ACM* , 48(2):274–296, March 2001.

[26] Maxim Sviridenko. An improved approximation algorithm for the metric uncapacitated facility location problem. *Integer Programming and Combinatorial Optimization: 9th International IPCO Conference Cambridge, MA, USA, May 27–29, 2002 Proceedings*, pages 240–257, 2002.

[27] Mohammad Mahdian, Yinyu Ye, and Jiawei Zhang. Approximation algorithms for metric facility location problems. *SIAM J. Comput.*, 36(2):411–432, 2006.

[28] Jaroslaw Byrka. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques: 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007. Proceedings*, pages 29–43, 2007.

[29] Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Automata, Languages and Programming: 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 77–88, 2011.

[30] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In STOC, pages 731–740, 2002.

[31] Sudipto Guha and Samir Khuller. Greedy strikes back: Improved facility location algorithms. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98, pages 649–657, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[32] Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996.

[33] Moses Charikar, Sudipto Guha, Éva Tardos, and David B Shmoys. A constant-factor approximation algorithm for the k-median problem. In *STOC*, pages 1–10. ACM, 1999.

[34] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004.

[35] Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. In *STOC*, pages 901–910, 2013.

[36] Jaroslaw Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k-median, and positive correlation in budgeted optimization. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA)*, pages 737–756, 2015.

[37] Jaroslaw Byrka, Krzysztof Fleszar, Bartosz Rybicki, and Joachim Spoerhase. Bi-factor approximation algorithms for hard capacitated k-median problems. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 722–736. SIAM, 2015.

[38] Shi Li. Approximating capacitated k-median with (1 + eps)k open facilities. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 786–796. SIAM, 2016.

[39] H. Gökalp Demirci and Shi Li. Constant approximation for capacitated k-median with $(1 + \epsilon)$-capacity violation. *CoRR*, abs/1603.02324, 2016.

[40] Jaroslaw Byrka, Bartosz Rybicki, and Sumedha Uniyal. An approximation algorithm for uniform capacitated k-median problem with $(1 + \epsilon)$ capacity violation. In *Integer Programming and Combinatorial Optimization - 18th International Conference, IPCO 2016, Liège, Belgium, June 1-3, 2016, Proceedings*, pages 262–274, 2016.

[41] Madhukar R. Korupolu, C. Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems. *J. Algorithms*, 37(1):146–188, 2000.

[42] Fabián A. Chudak and David P. Williamson. Improved approximation algorithms for capacitated facility location problems. *Math. Program.*, 102(2):207–222, 2005.

[43] Ankit Aggarwal, Anand Louis, Manisha Bansal, Naveen Garg, Neelima Gupta, Shubham Gupta, and Surabhi Jain. A 3-approximation algorithm for the facility location problem with uniform capacities. *Math. Program.*, 141(1-2):527–547, 2013.

[44] Martin Pál, Éva Tardos, and Tom Wexler. Facility location with nonuniform hard capacities. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 329–338, 2001.

[45] Manisha Bansal, Naveen Garg, and Neelima Gupta. A 5-approximation for capacitated facility location. In *Algorithms - ESA 2012 - 20th Annual European Symposium, Ljubljana, Slovenia, September 10-12, 2012. Proceedings*, pages 133–144, 2012.

[46] Hyung-Chan An, Mohit Singh, and Ola Svensson. Lp-based algorithms for capacitated facility location. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 256–265, 2014.

[47] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[48] Zoya Svitkina. Lower-bounded facility location. *ACM Trans. Algorithms*, 6(4):69:1–69:16, 2010.

[49] Chaitanya Swamy and David B. Shmoys. Fault-tolerant facility location. *ACM Trans. Algorithms*, 4(4):51:1–51:27, August 2008.

[50] Mohammadtaghi Hajiaghayi, Wei Hu, Jian Li, Shi Li, and Barna Saha. A constant factor approximation algorithm for fault-tolerant k-median. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 1–12. SIAM, 2014.

[51] Ravishankar Krishnaswamy, Amit Kumar, Viswanath Nagarajan, Yogish Sabharwal, and Barna Saha. The matroid median problem. In *Proceedings of the annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1117–1130. SIAM, 2011.

[52] Amit Kumar. Constant factor approximation algorithm for the knapsack median problem. In *Proceedings of the annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 824–832. SIAM, 2012.

[53] Moses Charikar and Shi Li. A dependent lp-rounding approach for the k-median problem. *Automata, Languages, and Programming (ICALP)*, pages 194–205, 2012.

[54] Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3):533–550, May 1986.

[55] Samir Khuller and Yoram J. Sussmann. The capacitated K-center problem. *SIAM J. Discrete Math.*, 13(3):403–418, 2000.

[56] Marek Cygan, MohammadTaghi Hajiaghayi, and Samir Khuller. LP rounding for k-centers with non-uniform hard capacities. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 273–282, 2012.

[57] Hyung-Chan An, Aditya Bhaskara, Chandra Chekuri, Shalmoli Gupta, Vivek Madan, and Ola Svensson. Centrality of trees for capacitated k-center. In *IPCO*, 2014.

[58] Samir Khuller, Robert Pless, and Yoram J. Sussmann. Fault tolerant k-center problems. *Theoretical Computer Science*, 242(12):237 – 245, 2000.

[59] Danny Z. Chen, Jian Li, Hongyu Liang, and Haitao Wang. Matroid and knapsack center problems. *Integer Programming and Combinatorial Optimization: 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings*, pages 110–122, 2013.

[60] Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 642–651, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[61] Richard Matthew McCutchen and Samir Khuller. Streaming algorithms for k-center clustering with outliers and with anonymity. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques, 11th International Workshop, APPROX 2008, and 12th International Workshop, RANDOM 2008, Boston, MA, USA, August 25-27, 2008. Proceedings*, pages 165–178, 2008.

[62] Gustavo Malkomes, Matt J. Kusner, Wenlin Chen, Kilian Q. Weinberger, and Benjamin Moseley. Fast distributed k-center clustering with outliers on massive data. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1063–1071, 2015.

[63] Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 826–835, 2008.

[64] Deeparnab Chakrabarty, Prachi Goyal, and Ravishankar Krishnaswamy. The non-uniform k-center problem. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55, pages 67:1–67:15, 2016.

[65] Sara Ahmadian and Chaitanya Swamy. Approximation algorithms for clustering problems with lower bounds and outliers. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 69:1–69:15, 2016.

[66] Marek Cygan and Tomasz Kociumaka. Constant factor approximation for capacitated k-center with outliers. In *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014), STACS 2014, March 5-8, 2014, Lyon, France*, pages 251–262, 2014.

[67] Babak Behsaz and Mohammad R. Salavatipour. On minimum sum of radii and diameters clustering. *Algorithmica*, 73(1):143–165, 2015.

[68] Matt Gibson, Gaurav Kanade, Erik Krohn, Imran A. Pirwani, and Kasturi R. Varadarajan. On metric clustering to minimize the sum of radii. *Algorithmica*, 57(3):484–498, 2010.

[69] Matt Gibson, Gaurav Kanade, Erik Krohn, Imran A. Pirwani, and Kasturi R. Varadarajan. On clustering to minimize the sum of radii. *SIAM J. Comput.*, 41(1):47–60, 2012.

[70] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 626–635, 1997.

[71] Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better streaming algorithms for clustering problems. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 30–39, 2003.

[72] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: An efficient clustering algorithm for large databases. *Inf. Syst.*, 26(1):35–58, 2001.

[73] Prabhakar Raghavan and Clark D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[74] Alexander A. Ageev and Maxim Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *J. Comb. Optim.*, 8(3):307–328, 2004.

[75] A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *FOCS*, pages 588–597, 2001.

[76] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. Dependent rounding and its applications to approximation algorithms. *Journal of the ACM*, 53(3):324–360, 2006.

[77] Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. When LP is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

[78] Jaroslaw Byrka, Aravind Srinivasan, and Chaitanya Swamy. Fault-tolerant facility location: A randomized dependent lp-rounding algorithm. In *Integer Programming and Combinatorial Optimization, 14th International Conference, IPCO 2010, Lausanne, Switzerland, June 9-11, 2010. Proceedings*, pages 244–257, 2010.

[79] V. S. Anil Kumar, Madhav V. Marathe, Srinivasan Parthasarathy, and Aravind Srinivasan. A unified approach to scheduling on unrelated parallel machines. *J. ACM*, 56(5), 2009.

[80] Benjamin Doerr. Nonindependent randomized rounding and an application to digital halftoning. *SIAM J. Comput.*, 34(2):299–317, 2004.

[81] Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.

[82] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Multi-budgeted matchings and matroid intersection via dependent rounding. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1080–1097, 2011.

[83] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 575–584, 2010.

[84] Nicholas J. A. Harvey and Neil Olver. Pipage rounding, pessimistic estimators and matrix concentration. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 926–945, 2014.

[85] D. Harris, T. Pensyl, A. Srinivasan, and K. Trinh. Fairness in resource allocation and slowed-down dependent rounding. Technical report, University of Maryland, 2016.

[86] Alessandro Panconesi and Aravind Srinivasan. Randomized distributed edge coloring via an extension of the chernoff-hoeffding bounds. *SIAM J. Comput.*, 26(2):350–368, 1997.

[87] Fabián A. Chudak and David B. Shmoys. Improved approximation algorithms for the uncapacitated facility location problem. *SIAM J. Comput.*, 33(1):1–25, 2003.

[88] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

[89] Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[90] Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293 – 306, 1985.

[91] Dorit S Hochbaum and David B Shmoys. A best possible heuristic for the k-center problem. *Mathematics of operations research*, 10(2):180–184, 1985.

[92] Dorit S. Hochbaum, editor. *Approximation Algorithms for NP-hard Problems*. PWS Publishing Co., Boston, MA, USA, 1997.