Uniwersytet Wrocławski

Instytut Informatyki

Algorytmy aproksymacyjne dla klastrowania i submodularnych problemów lokalizacji obiektów

Fateme Abbasi

Rozprawa doktorska

Napsiana pod kierunkiem
Prof. dr hab. Jarosława Byrki

Wrocław, 2025

University of Wrocław

Institute of Computer Science

Approximation Algorithms for Clustering and Submodular Facility Location Problems

Fateme Abbasi

A dissertation supervised by
Prof. dr hab. Jarosław Byrka

Submitted for a degree of
Doctor of Philosophy

Wrocław, 2025

Abstrakt

W niniejszej pracy rozważamy dwa fundamentalne wyzwania w teoretycznej informatyce —grupowanie i lokalizację obiektów — które były badane przez wiele lat, ale wciąż pozostają nierozwiązane w wielu aspektach. Dzięki zastosowaniom w takich dziedzinach jak optymalizacja, uczenie maszynowe, eksploracja danych oraz badania operacyjne, dalsze badania w tych obszarach są niezwykle wartościowe. W klasie problemów grupowania koncentrujemy się na modelach opartych na centrach (ang. center-based models), natomiast w obszarze lokalizacji obiektów badamy submodularny problem lokalizacji obiektów, będący rozszerzoną wersją tradycyjnego problemu lokalizacji obiektów. Ponieważ większość problemów w tych obszarach jest NP-trudna, skupiamy się na projektowaniu algorytmów aproksymacyjnych dla tych problemów.

Ogólnym celem grupowania jest podział danego zbioru danych na podobne grupy. W przypadku k-grupowania dążymy do podziału zbioru danych na k odrębnych klastrów przy minimalizacji funkcji celu, która zazwyczaj stanowi wektor odległości pomiędzy punktami danych a centrami ich odpowiednich klastrów. Projektujemy parametryzowane algorytmy aproksymacyjne dla różnych funkcji celu w problemach k-grupowania. Po pierwsze, rozważamy Socially Fair (k, z)-Clustering i przedstawiamy nowe wyniki dotyczące parametryzowanej aproksymowalności tego problemu w przestrzeniach geometrycznych, szczególnie w wysokowymiarowych przestrzeniach Euklidesowych. Następnie rozważamy reżim algorytmiczny projektowania $(1 + \epsilon)$-aproksymacyjnego algorytmu dla problemu normowego k-grupowania, który działa w czasie $f(k, \epsilon)\mathsf{poly}(n)$ (często nazywanego efektywnym parametryzowanym schematem aproksymacyjnym, EPAS). Prezentujemy ujednolicony EPAS, który działa dla różnych funkcji celu i przestrzeni metrycznych, wprowadzając nową koncepcję zwaną ϵ-scatter dimension — miarą złożoności przestrzeni metrycznej. Dla dowolnej funkcji celu w przestrzeni metrycznej M uzyskujemy EPAS, pod warunkiem, że funkcja celu jest monotoniczną normą, a ϵ-scatter dimension przestrzeni M jest ograniczone przez funkcję zależną od ϵ.

Następnie rozważamy Submodularny Problem Lokalizacji Obiektów (SFL), w którym mamy dany zbiór klientów, zbiór obiektów oraz koszt otwarcia każdego obiektu określony przez monotoniczną funkcję submodularną. Celem jest przypisanie każdego klienta do obiektu w taki sposób, aby zminimalizować łączny koszt przypisania i otwarcia obiektów. Dokonujemy postępu w rozwiązaniu pytania — czy SFL dopuszcza aproksymację o stałym współczynniku — prezentując algorytm o współczynniku aproksymacji $\log \log n$ dla SFL. Nasze podejście jest elastyczne i rozszerza się na uogólnienia oraz warianty SFL. Dodatkowo uzyskujemy ulepszony algorytm aproksymacyjny dla problemu uniwersalnej stochastycznej lokalizacji obiektów.

Abstract

In this thesis we consider two fundamental challenges in theoretical computer science—clustering and facility location—which have been studied for many years yet remain unsolved in many aspects. With applications in areas like optimization, machine learning, data mining, and operations research, further research in these areas is highly valuable. In the class of clustering problems, we focus on center-based models while in the area of facility location, we study submodular facility location problem, an extended version of traditional facility location problem. Given that most of the problems in these areas are NP-hard, we focus on designing approximation algorithms for the problems.

The general goal of clustering problems is to divide a given dataset into similar groups. In k-clustering, we aim to divide a given dataset to k distinct clusters while minimizing the objective function, specifically a vector of distances between the data points and the centers of their corresponding clusters. We design parameterized approximation algorithms for various k-clustering objective functions. First, we consider Socially Fair (k, z)-Clustering, and provide a new results on parameterized approximability of the problem in geometric spaces, particularly in high-dimensional discrete Euclidean spaces. Then, we consider the algorithmic regime of designing a $(1+\epsilon)$-approximation algorithm for a norm k-clustering problem that runs in time $f(k, \epsilon)\mathsf{poly}(n)$ (sometimes called an efficient parameterized approximation scheme or EPAS). We present a unified EPAS that works over various objective functions and several metric spaces by introducing a novel concept called bounded ϵ-scatter dimension —an intrinsic complexity measure of a metric space that relaxes the standard notion of bounded doubling dimension. For any clustering objective in metric space M we obtain an EPAS, as long as the objective is a monotone norm and the ϵ-scatter dimension of M is upper bounded by a function of ϵ.

Next, we consider Submodular Facility Location (SFL), where we are given a set of clients, a set of facilities, and an opening cost for each facility defined by a monotone submodular function. The goal is to assign each client to a facility while minimizing the combined cost of assignment and facility opening. We make progress toward a longstanding open question—whether SFL admits a constant-factor approximation—by presenting a $\log \log n$-approximation algorithm for SFL. Our approach is flexible and extends to generalizations and variants of SFL. Additionally, we obtain an improved approximation algorithm for the related Universal Stochastic Facility Location problem.

Acknowledgments

First and foremost, I would like to express my deepest thanks to my advisor, Jarosław Byrka, for his guidance, support, and constant encouragement throughout my PhD. His patience, high standards, and dedication have challenged me to grow as a researcher, writer, and presenter. I truly appreciate the countless hours he spent reviewing my work and offering thoughtful advice. This dissertation would not have been possible without his support, and I am incredibly grateful for everything he has taught me.

I sincerely thank my co-authors—Marek Adamczyk, Sandip Banerjee, Miguel Bosch-Calvo, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Fabrizio Grandoni, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, Krzysztof Sornat, Joachim Spoerhase, and Antoine Tinguely. Their collaboration and insights were instrumental in shaping these works, and I feel incredibly fortunate to have had the opportunity to work alongside such brilliant researchers. A special thanks to Joachim, Kamyar, Ameet, and Sandip for creating a pleasant, supportive, and collaborative research environment, as well as for their guidance and invaluable advice. I am truly grateful for everything I have learned from them.

Beyond my co-authors, I would like to thank Martin Böhm, Marek Chrobak, Nathan Klein, and Jan Marcinkowski for their insightful discussions on interesting problems— I appreciate the opportunity to learn from them.

I am endlessly grateful to my family and friends for their support and encouragement. My deepest appreciation goes to my parents, sisters, and brothers, who have been a constant source of strength, support, and motivation—for that, I am forever grateful. I also want to thank my dearest friends—Neda, Behzad, Yasmin, Arash, Zahra, Mohammad, Aida, and Sadia—as well as all my friends in Wrocław for their kindness, encouragement, and for making me feel at home even when I was far away.

Finally, I want to thank my colleagues at the institute—Marcin Bieńkowski, Łukasz Jeż, Artur Kraska, and Yongho Shin—as well as all my other colleagues for creating a warm and inspiring work environment.

Contents

Chapter 1

Introduction

This thesis explores Clustering, an algorithmic task that involves grouping similar points to optimize certain cost functions. The objective function in clustering often focuses on minimizing the distances between points within the same group. Clustering has received significant attention across various research domains, including optimization, data mining, machine learning, and computational geometry.

In clustering, typically we are given a set of points, and the goal is to group similar points, where the definition of similarity is determined by the chosen objective function. For example, in center-based models, the goal is to find k centers so that the points in each group or cluster are close enough to their center based on distances. Center-based clustering problems, such as k-Center, k-Median, and k-Means, have been studied for more than half a century, yet they continue to present challenging open questions, keeping this an active area of research.

This thesis is based on the following papers:

1. Fateme Abbasi, Sandip Banerjee, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. **Parameterized Approximation For Robust Clustering in Discrete Geometric Spaces.** 51st International Colloquium on Automata, Languages, and Programming (ICALP 2024) [2]

2. Fateme Abbasi, Sandip Banerjee, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. **Parameterized approximation schemes for clustering with general norm objectives**. In 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS 2023) [3]

3. Fateme Abbasi, Marek Adamczyk, Miguel Bosch-Calvo, Jarosław Byrka, Fabrizio Grandoni, Krzysztof Sornat, and Antoine Tinguely. **An O(loglog n)-Approximation for Submodular Facility Location.** 51st International Colloquium on Automata, Languages, and Programming (ICALP 2024) [1]

We present new results in parameterized approximation schemes and approximation algorithms for clustering and Submodular Facility Location problems. Chapter 2, presents results from [2] where we achieve three distinct results, including an improved constant-factor approximation for a socially fair clustering problem. Chapter 3 addresses results from [3], where we develop a new framework to achieve EPAS for norm k-clustering problems. Finally in Chapter 4, we discuss the results from [1] offering improved approximation algorithm for Submodular Facility Location Problem.

1.1 Approximation Algorithms

Due to the NP-hard nature of clustering problems, approximation algorithms are crucial to find feasible solutions. These algorithms seek near-optimal solutions within a reasonable computational time, balancing computational efficiency and accuracy. We consider two main types of approximation algorithm:

- **Traditional Approximation Algorithms**: These algorithms provide solutions in polynomial time, ensuring that the objective value $f(x)$ of a solution x is within a factor α of the optimal $f(o)$. The α is an approximation factor that determines how close the approximation is to the ideal solution. 0
- **Parameterized Approximation Algorithms**: Building on the traditional approach, these algorithms incorporate an additional parameter k, offering a more nuanced solution. They aim to achieve solutions with a complexity that is polynomial in input size and a function of k. This parameter typically reflects a specific characteristic of the input, and the algorithms are effective when k is small. They balance the solution quality of traditional algorithms with the efficiency of fixed-parameter tractable (FPT) algorithms, offering solutions in $f(k)n^{O(1)}$ time, where $f(k)$ is independent of the input size n.

1.1.1 Types of Schemes for Parameterized Approximation Algorithms

A parameterized approximation algorithm aims to find approximate solutions to NP-hard optimization problems in polynomial time in the input size and a function of a specific parameter. These algorithms combine the best aspects of traditional approximation algorithms and fixed-parameter tractability (FPT). To further capture the effectiveness and efficiency of approximation algorithms, several advanced schemes have been developed.

- **Polynomial-Time Approximation Scheme (PTAS)**: A PTAS is an algorithm that, for any given $\varepsilon > 0$, produces a solution within a factor $1 + \varepsilon$ of the optimal solution. The running time of a PTAS is polynomial in the input size for any fixed ε, but may vary significantly with different values of ε. For instance, an algorithm running in $O(n^{1/\varepsilon})$ or $O(n^{\exp(1/\varepsilon)})$ is considered a PTAS.
- **Efficient Polynomial-Time Approximation Scheme (EPTAS)**: One issue with PTAS algorithms is that their runtime exponent can increase dramatically as ε decreases. An EPTAS addresses this by ensuring the runtime is $O(n^c)$ for a constant c that is independent of ε, thus maintaining efficiency

regardless of the ε value. However, the constant under the big-O notation may still depend on ε. Essentially, an EPTAS operates in fixed-parameter tractable (FPT) time with ε as the parameter.

- **Parameterized Approximation Scheme (PAS)**: For any $\varepsilon > 0$, a $(1+\varepsilon)$-approximation can be computed in $f(k,\varepsilon)n^{g(\varepsilon)}$ time for some functions f and g. This approach circumvents the lower bounds in terms of polynomial-time approximation and fixed-parameter tractability, enabling it to address problems that are APX-hard and W[1]-hard. A PAS is similar in spirit to a polynomial-time approximation scheme (PTAS) but additionally exploits a given parameter k. Since the degree of the polynomial in the runtime of a PAS depends on a function $g(\varepsilon)$, the value of ε is assumed to be arbitrary but constant in order for the PAS to run in FPT time.

- **Efficient Parameterized Approximation Scheme (EPAS)**: Similar to an EPTAS, an EPAS ensures that the runtime remains efficient with respect to both the input size and the parameter ε. For any $\varepsilon > 0$, an EPAS computes a $(1+\varepsilon)$-approximation in $f(k,\varepsilon)n^{O(1)}$ time, maintaining efficiency and practicality even as ε changes.

These advanced schemes offer a blend of precision and efficiency, making them suitable for tackling various NP-hard optimization problems where traditional methods fall short.

1.1.2 Parameterized Complexity

Parameterized complexity theory provides a refined analysis of computational problems by incorporating additional parameters into the problem's input. Unlike classical complexity theory, where the complexity is measured by the input size, parameterized complexity introduces a *parameter* to classify problems based on their complexity as a function of both the input size and the parameter. This approach allows for a finer classification of computational problems, particularly NP-hard ones.

Formally, a parameterized problem is a pair (L,κ), where $L \subseteq \Sigma^*$ is a language over a finite alphabet Σ, and $\kappa : \Sigma^* \to \mathbb{N}$ is a parameterization, a polynomial-time computable function mapping each input to a non-negative integer. The function $\kappa(x)$ is referred to as the *parameter* of the input x. A parameterized problem is thus a classical decision problem augmented by a parameter that often captures some structural aspect of the problem instance.

A classical example is the well-known NP-complete problem Vertex Cover:

Vertex Cover
Instance: A graph $G = (V,E)$ and an integer $k \in \mathbb{N}$.
Problem: Determine whether G has a vertex cover of size at most k.

The parameterized version of this problem, called *k-Vertex Cover*, is defined by setting the parameter $\kappa(G,k) = k$:

k-Vertex Cover
Instance: A graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Parameter: k.
Problem: Determine whether G has a vertex cover of size at most k.

Here, the parameter k represents the size of the solution. When a problem's parameter corresponds to a structural or solution-related feature, such as the size of a solution, it is called a *natural parameter*. A fundamental notion in parameterized complexity is *fixed-parameter tractability*, which characterizes problems that are efficiently solvable when the parameter is small, even if the problem is NP-hard in the classical sense.

Fixed-Parameter Tractability

A parameterized problem (L, κ) is said to be *fixed-parameter tractable* (FPT) if there exists an algorithm A, a constant c, and a computable function $f : \mathbb{N} \to \mathbb{N}$, such that for any instance $x \in \Sigma^*$ with parameter $k = \kappa(x)$, the algorithm A decides whether $x \in L$ in time $f(k) \cdot |x|^c$. The class of all fixed-parameter tractable problems is denoted by FPT.

Intuitively, a problem is in FPT if its computational complexity grows relatively modestly with respect to the input size, but may depend arbitrarily on the parameter k. This framework is useful for dealing with NP-hard problems when the parameter is fixed or small.

Once the notion of FPT is established, the next important concept is *FPT-reductions*, which preserve fixed-parameter tractability across problem transformations.

Definition 1.1.1 (FPT-reduction)**.** *An* FPT-reduction *from a parameterized problem $Q \subseteq \Sigma^* \times \mathbb{N}$ to another parameterized problem $Q' \subseteq \Sigma^* \times \mathbb{N}$ is a computable function $R : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ such that for all instances (x, k):*

- $(x, k) \in Q \iff R(x, k) \in Q'$,
- *$R(x, k)$ runs in time $f(k) \cdot |x|^c$ for some computable function f and constant c,*
- *if $R(x, k) = (x', k')$, then $k' \leq g(k)$ for some computable function g.*

FPT-reductions allow the transfer of fixed-parameter tractability between problems. They are central to classifying problems into parameterized complexity classes, such as W[1], which is widely believed to be the parameterized analog of NP.

Class W[1]: A parameterized problem (L, κ) belongs to the class W[1] if there is an FPT-reduction from (L, κ) to the *k-CLIQUE* problem. A problem is W[1]-hard if every problem in W[1] can be FPT-reduced to it.

It is conjectured that problems in W[1], including *k-CLIQUE* are not fixed-parameter tractable, much like NP-complete problems are unlikely to be solvable in polynomial time. A deeper exploration of parameterized complexity, along with the W-hierarchy, can be found in foundational works such as [62], and [52].

1.2 Problem Setting

In this thesis, we explore various classes of metrics and clustering objective functions. First we discuss the objective functions considered in this thesis.

- (k, z)-**Clustering**: In the context of the k-**clustering** problem, we are given a set of n data points P, a set of potential centers defined as F, within a metric space defined as $M = (P \cup F, \delta)$. Additionally, a positive integer k and an objective function $f : \mathbb{R}^P \to \mathbb{R}$. The task involves selecting a set of k "open" centers denoted as $X \subseteq F$, which subsequently generates a distance vector $\boldsymbol{\delta}(P, X) = (\delta(p, X))_{p \in P}$, where $\delta(p, X) = \min_{x \in X} \delta(p, x)$ represents the distance from point p to the closest center in X. The goal is to minimize $f(\boldsymbol{\delta}(P, X))$.

  Consider the metric space $M = (P \cup F, \delta)$, where P and F are disjoint sets and δ represents the distance function. Let $k > 0$ and $z > 0$ be positive integers. In each of the following clustering objectives, the goal is to find k subsets of centers $X \subseteq F$ that minimize:

$$\begin{aligned} k\text{-Median} : &\quad \sum_{p \in P} \delta(p, X), \\ k\text{-Means} : &\quad \sum_{p \in P} \delta(p, X)^2, \\ k\text{-Center} : &\quad \max_{p \in P} \delta(p, X), \\ (k, z)\text{-Clustering} : &\quad \sum_{p \in P} \delta(p, X)^z. \end{aligned}$$

- **Socially Fair** k**-Median**: Variants like Socially Fair k-Median (or Robust k-Median) extends clustering objectives to cater to fairness considerations and uncertain data scenarios, respectively, offering versatile tools to address diverse clustering challenges. In Socially Fair k-Median, along with the point set P, we are given m different (not necessarily disjoint) subgroups such that $P = \bigcup_{i \in [m]} P_i$. Our goal is to find a set X of centers that incurs fair costs to the groups by minimizing the maximum cost over all the groups. In other words,

$$\min_{\substack{X \subseteq F \\ |X| = k}} \max_{i \in [m]} \sum_{p \in P_i} \delta(p, X) \,.$$

- **Socially Fair** (k, z)**-Clustering**, which generalizes Socially Fair k-Median, the objective function is as follows:

$$\min_{\substack{X \subseteq F \\ |X| = k}} \max_{i \in [m]} \sum_{p \in P_i} \delta(p, X)^z \,.$$

- **Facility Location**[1]: In the Facility Location problem, denoted as FL, we are given a set C of n

[1]Note that the notation used to define Facility Location problem[1] in Chapter 4 differs from that in other chapters.

clients and a set F of m facilities ,with a metric distance $d : (C \cup F) \times (C \cup F) \to \mathbb{R}_{\geq 0}$. We aim to minimize the sum of the distances from each client to the corresponding facility plus the total opening cost of the facility, in other words, the goal is to minimize:

$$\sum_{c \in C} d(c, \varphi(C)) + \sum_{f \in F} f \cdot y_f$$

Here, $\varphi : C \to F$ represents the assignment function of each c to some facility, and $y_f \in \{0, 1\}$ is a binary decision variable indicating whether facility f is open ($y_f = 1$) or closed ($y_f = 0$).

- **Submodular Facility Location** extends classical Facility Location problem by incorporating submodular costs associated with serving sets of clients by individual facilities. In Submodular Facility Location (SFL) we are given a monotone submodular[2] (opening cost) function $g : 2^c \to \mathbb{R}_{\geq 0}$ with $g(\emptyset) = 0$. Notice that $g(\cdot)$ is non-negative. The goal is to minimize:

$$\sum_{c \in C} d(c, \varphi(C)) + \sum_{f \in F} g(\varphi^{-1}(f)).$$

Next, we discuss the metric spaces considered in this Thesis.

- **High-dimensional Euclidean spaces**: These metrics involve measuring distances in spaces characterized by a large number of dimensions, where the distance between any two points is calculated using the Euclidean distance formula. We consider both continuous and discrete Euclidean spaces.
  - **Continuous Euclidean Spaces:** In this case, for the problems mentioned earlier, we are allowed to choose centers from the (high-dimensional) continuous Euclidean space, denoted as $F = \mathbb{R}^d$. The set $P \subseteq \mathbb{R}^d$ is a finite set of points.
  - **Discrete Euclidean Spaces:** Here, both sets P and F are finite subsets of $\mathbb{R}^d$, i.e., $P, F \subseteq \mathbb{R}^d$.
- **Metrics of bounded doubling dimension**: The doubling dimension of a metric space (M, δ) is the smallest integer d such that any ball of radius $2r$ can be covered by 2^d balls of radius r [85, 80].
- **Graph Metric:** In the case of graph metric, we are given a (weighted) graph $G = (V, E)$ and the metric δ_G on V as the shortest path metric, i.e., $\delta_G(u, v)$ is the shortest distance of a path connecting u and v.
- **Planar metrics**: Planar metrics are defined on graphs that can be embedded in the plane without any edge crossings. Distances in planar metrics are typically calculated using the shortest path in the embedded graph.

[2]We recall that $g(\cdot)$ is submodular iff, for every $S, T \subseteq C$, $g(S) + g(T) \geq g(S \cap T) + g(S \cup T)$. The function is also monotone if $g(T) \leq g(S)$ for every $T \subseteq S \subseteq C$. As usual in this framework, we assume to have an oracle access to $g(\cdot)$: given $S \subseteq C$, we can obtain the value of $g(S)$ in polynomial time.

- **Bounded treewidth metrics**: These metrics are defined on graphs with bounded treewidth. The treewidth of a graph G is the smallest integer t such that there exists a tree decomposition with maximum size bag $t + 1$. A tree decomposition of a graph $G = (V, E)$ is defined as a tree T with node set V, where each node, called a bag, is a subset of V satisfying the following conditions: (i): The union of all bags equals the vertex set V. (ii) For every vertex $u \in V$, the bags containing u form a connected component in T. (iii) For every edge $(u, v) \in E$, there exists a bag S in the tree decomposition such that both u and v are contained in S.

1.3 Thesis Outline of Results

1.3.1 Socially Fair/Robust Clustering in Discrete Euclidean space

In Chapter 2, we study Socially Fair (k, z)-Clustering in a discrete Euclidean setting, extending the classic k-Median, k-Means, and k-Center problems. The objective is to cluster n weighted points, each belonging to one (or more) of m groups, and minimize the maximum weighted distance between these points and k chosen centers across all groups. This approach is particularly useful in managing uncertainties in input data, as real datasets often lack precision or completeness. To address such challenges, Anthony et al. [11] introduced a robustness framework for the k-Median problem under uncertain demand. In this setting, we are given multiple groups of points, and the goal is to select k centers that perform well across all groups. In other words, the objective is to minimize the worst-case connection cost over all point groups. They call this problem Robust k-Median. A similar objective "fairness" has also been studied, where the goal is to develop solutions that is fair for all specified groups. This concept of fairness in clustering has been formalized by Abbasi et al. [4] and Ghadiri et al. [69] under the Socially Fair k-Median model. In this model, Abbasi et al. address fairness by weighting clients inversely to group size, while Anthony et al. focus on unweighted robust clustering. Notably, while these two problems are mathematically equivalent, they arise from entirely different contexts. By allowing for arbitrary point(client) weights our model captures both of these settings.

In general metric, there exists a polynomial $O\left(\frac{\log n}{\log \log n}\right)$ approximation, that is tight even in the line metric [103]. Building on this recent work has focused on designing constant-factor parameterized (FPT) approximation algorithms. Goyal and Jaiswal [73] design a FPT-approximation algorithms with ratio $(3^z + \epsilon)$ for Robust (k, z)-Clustering that is tight. Motivated by the the lower bound of $(3^z - o(1))$ for general discrete metric spaces [73], we explore whether the geometric properties of discrete high-dimensional space can be utilized by FPT approximation algorithms for Robust (k, z)-Clustering, potentially bypassing the $(3^z - o(1))$ lower bound. In this thesis, we answer the above question by designing an FPT approximation algorithm with a ratio slightly below the barrier of 3^z. Our algorithm is a refined adaptation of FPT algorithm by Goyal and Jaiswal[73]. While the core structure remains similar, we introduce a slight modification that takes advantage of geometric properties in the Euclidean setting. This adjustment allows us to overcome certain limitations of the original approach, leading to an improved approximation ratio. A key aspect of Goyal and Jaiswal's analysis is a basic projection property of metric spaces. We show that, with slight

additional assumptions, this property can be strengthened in the Euclidean setting. To achieve this, we expand the search space by introducing midpoint closure—specifically, by adding the closets facility to the midpoints of pairs of facilities to our space. The analysis requires several new ideas and enables us to achieve a $(3^z - \epsilon)$-approximation algorithm in FPT time. More formally, we prove the following theorem.

Theorem 1.3.1 (High-Dimensional Euclidean Space)**.** *There exists a universal constant $\eta_0 > 0.0006$ such that for any constant positive integer z, there is a factor $3^z(1-\eta_0)$ FPT approximation algorithm for* Socially Fair (k,z)-Clustering *in discrete Euclidean space $\mathbb{R}^d$ that runs in time $2^{O(k \log k)}\mathsf{poly}(m,n,d)$.*

We want to highlight that our algorithm's running time is polynomially dependent on d. The main takeaway from Theorem 1.3.1 is that we can actually improve the factor of 3^z. This result shows that, in the context of FPT, geometric spaces are easier to work with for Robust (k,z)-Clustering than general metric spaces, that is different from polynomial-time approximation algorithms, where the problem is equally hard in both general and Euclidean metrics [21].

Next, we focus on providing a complete characterization of the existence of an EPAS in discrete Euclidean spaces. We like to remark that while an EPAS exists in continuous Euclidean spaces of any dimension and in discrete Euclidean spaces of dimension $o(\log \log n)$ [3] (we discuss it in Chapter 3), determining the existence of EPAS in discrete spaces of dimension $\Omega(\log \log n)$ remains an open challenge. We prove that even the special case of the k-Center problem does not admit an EPAS in these settings, by constructing an instance of the discrete k-Center problem from an instance of the Multi-Colored Independent Set problem, which is known to be W[1]-hard.

In the next theorem, we prove that even the special case of k-Center does not admit an EPAS. This hardness holds for any ℓ_q metric and even in dimension $O(k \log n)$.

Theorem 1.3.2 (Hardness in Discrete Euclidean Space)**.** *For any constant positive integer q and any positive constant $\eta > 0$, there exists a function $d(k,n) = O(k \log n)$ such that there is no factor-$(3/2-\eta)^{1/q}$ FPT approximation algorithm for the discrete k-Center problem in $\mathbb{R}^{d(k,n)}$ under the ℓ_q metric unless $\mathsf{W}[1] = \mathsf{FPT}$. Moreover, for the ℓ_2 metric this hardness holds even for some dimension $O(\log n)$, that is, independently of k.*

This result draws attention to the differences between discrete and continuous settings in high-dimensional Euclidean spaces, a topic that has garnered significant attention in recent years [46, 42, 43]. As previously noted, the continuous setting does allow for an EPAS [3], which implies that Socially Fair (k,z)-Clustering is more difficult in dicerte setting.

This is different from the results of Cohen-Addad et al. [43], who showed that the continuous versions of k-Median and k-Means in geometric spaces are harder to approximate in polynomial time compared to their discrete versions.

Additionally, we complete the FPT-approximability landscape by designing an EPAS for the problem in doubling metrics when the dimension is $d = o_k(\log n)$[3]. Notably, the doubling dimension of the d-dimensional discrete Euclidean metric is $\Theta(d)$, which means we achieve an EPAS for discrete Euclidean

[3]We use the notation $o_k(\cdot)$ to hide multiplicative factors depending only on k.

spaces of dimension $o_k(\log n)$.

Our algorithm consists of two main components: instance compression and decomposition of the doubling metric into smaller balls. While these are standard techniques, applying them directly to Robust (k, z)-Clustering faces challenges. A natural approach is to reduce the number of groups and apply (k, z)-Clustering coreset [48] to further compress the instance. However, due to the lower bounds on coresets in Euclidean space, this approach results in a doubly exponential running time in the dimension d, preventing an efficient polynomial-time approximation scheme (EPAS). Alternative methods, such as reducing the number of points instead of groups, face difficulties in preserving cost approximations across groups. To address these issues, we adopt a more flexible and general definition of groups, allowing points to belong to multiple groups with different weights. This redefinition enables us to compress the instance while approximately preserving group costs. By integrating this approach with a standard ball decomposition technique for doubling metrics, we construct a coreset which can be used to design an EPAS for metrics of sub-logarithmic doubling dimension. The formal statement of the result is as follows:

Theorem 1.3.3 (EPAS for Doubling Metric of Sub-Logarithmic Dimension)**.** *There is an algorithm that computes* $(1+\epsilon)$*-approximate solution, for every* $\epsilon > 0$*, for Socially Fair* (k, z)*-Clustering in the metric of doubling dimension* d *in time* $f(k, d, \epsilon, z)\mathsf{poly}(m, n)$*, where* $f(k, d, \epsilon, z) = \left(\left(\frac{2^z}{\epsilon}\right)^d k \log k\right)^{O(k)}$.

The table below presents a comprehensive summary of the results for the problem in the discrete high-dimensional Euclidean setting.

		$\mathbb{R}^d$ **(discrete)**	**General**
P	Upper	$O(\frac{\log n}{\log\log n})$	[103]
	Lower	$\Omega(\frac{\log n}{\log\log n})$	[21]
FPT	Upper	$0.9994 \cdot 3^z$*	$(3^z + o(1))$[73]
	Lower	$(\sqrt{3/2} - o(1))$*	$(3^z - o(1))$ [73]

Table 1.1: New results are marked by *.

1.3.2 EPAS for General Norm Clustering

In Chapter 4, we explore k-clustering with a general norm objective function. In a general metric space, achieving Efficient Parameterized Approximation Schemes (EPAS)—an algorithm offering a $(1+\epsilon)$ approximation and running in time $f(k, \epsilon)poly(n)$ for every $\epsilon > 0$ is unattainable for even basic clustering problems. Consequently, previous researches have concentrated on developing algorithms for structured metric spaces, like planar graphs or Euclidean spaces. In the realm of continuous high-dimensional Euclidean spaces, EPASes represent the most efficient approximation schemes conceivable, as shown in [53, 12]. This has led to significant attention on EPASes for clustering problems over the last two decades, as shown by studies [83, 104, 108, 55, 20, 25, 47]. Our approach to Norm k-Clustering introduces a unified framework that is remarkably agnostic to the specifics of the underlying metric space. Despite this generality, it effectively

harnesses the intrinsic structure of various metrics, enabling the resolution of multiple EPASes across diverse settings. This includes previously unresolved clustering problems such as Ordered k-Median and (z,q)-Fair Clustering. Unlike traditional methods, which often rely on coreset construction or require tailored designs for each specific metric and objective, our framework is versatile and comprehensive. It encompasses nearly all center-based clustering objectives while also extending to new generalizations, thereby addressing numerous long-standing clustering challenges in a seamless and elegant manner. The tables below summarize the known results, including PTAS, EPAS, and polynomial time algorithms (P), for the objective functions discussed in this chapter. Entries marked with * denote new EPAS results presented in this work.

	$\mathbb{R}^d$ **(low)**		$\mathbb{R}^d$ **(high)**	
	PTAS	EPAS	PTAS	EPAS
k-Median	$O\left(2^{O\left(\left(\frac{\log(1/\epsilon)}{\epsilon}\right)^{d-1}\right)} n \log^{d+6} n\right)$[92]	$O\left(2^{(k/\epsilon)^{O(1)}} dn\right)$[94]	-	$O\left(2^{(k/\epsilon)^{O(1)}} dn\right)$[94]
k-Means	$n \cdot k \cdot (\log n)^{(d\epsilon^{-1})^{O(d)}}$ [44, 32, 65]	$O\left(2^{(k/\epsilon)^{O(1)}} dn\right)$[94]	APX-hard [12]	$O\left(2^{(k/\epsilon)^{O(1)}} dn\right)$[94]
k-Center	-	[84]	-	[84]
ℓ-Centrum	-	$O_{\epsilon,d}(k^2)$[24]	-	*
Priority k-Center	-	$O_{\epsilon,d}(k^2 log^2 n)$[24]	-	*
Ordered k-Median	-	$O_{\epsilon,d}(k^2 log^2 n)$[24]	-	*
(z,q)-Fair Clustering	-	*	-	*

Table 1.2: Summary of results for EPAS and PTAS in low- and high-dimensional Euclidean spaces. EPAS for k-Center on the planar metric is implied by a bicriteria EPTAS provided in [64] (improving upon a bicriteria PTAS [57]). Some results (e.g., [24]) are derived from the well-established concept of corsets.

	Doubling		**General**	
	PTAS	EPAS	P	FPT
k-Median	[118]	$\tilde{O}(2^{(1/\epsilon)^{O(d^2)}} n)$[36]	2.406 [33]	$(1+2/e+\epsilon)$ [39]
k-Means	$d^{O(d)} \cdot \epsilon^{O((-d/\epsilon))}$ [65]	$\tilde{O}(2^{(1/\epsilon)^{O(d^2)}} n)$[36]	5.912[33]	$(1+8/e+\epsilon)$ [39]
k-Center	-	$(k^k/\epsilon^{O(kd)}) \cdot n^{O(1)}$[60]	2 [71]	[17]
ℓ-Centrum	-	*	-	-
Priority k-Center	-	*	-	-
Ordered k-Median	-	*	-	-
(z,q)-Fair Clustering	-	*	-	$(3^z+\epsilon)$[73]

Table 1.3: Summary of results for running time in doubling metrics and approximation ratio algorithms in general metric.

	Bounded Treewidth		**Planar Graph**	
	PTAS	EPAS	PTAS	EPAS
k-Median	-	$\tilde{O}(\frac{k^3}{\epsilon^2}) \cdot TW(G)$[15]	[44]	$O\left(\Gamma \cdot k(\log^2 + \frac{\log k}{\epsilon^4} k)\right)$ [48]
k-Means	-	$O(\Gamma \cdot k(t + \log k))$[48]	[44]	$O\left(\Gamma \cdot k(\log^2 + \frac{\log k}{\epsilon^4} k)\right)$ [48]
k-Center	-	[91]	[57]	[64]
ℓ-Centrum	-	*	-	*
Priority k-Center	-	*	-	*
Ordered k-Median	-	*	-	*
(z, q)-Fair Clustering	-	*	-	*

Table 1.4: Summary of results for EPAS and PTAS in treewidth and planer graph. Some results (e.g., [15, 48]) are derived from the well-established concept of corsets. The notation Γ refers to $\Gamma = \min\left(\epsilon^{-2} + \epsilon^{-z},\ k\epsilon^{-2}\right)$.

Our main results are encapsulated in the following theorem.

Theorem 1.3.4. *Let f be an efficiently (approximately) computable monotone norm cost function. Then the k-clustering problem with cost function f admits an EPAS for the following input metrics: (i) metrics of bounded doubling dimension, (ii) continuous Euclidean spaces of any dimension, (iii) bounded treewidth metrics, and (iv) planar metrics.*

We highlight that our framework also applies to *asymmetric* norms, though it's worth noting that *asymmetric* norms can complicate the problem. For example, a poly-time $O(1)$-approximation algorithm exists for symmetric norms [29] but the asymmetric norm makes it $\Omega(\log n/\log\log n)$-hard to approximate even for the special case of Robust k-Median on the line metrics [21].

As we mentioned before, by *continuous* Euclidean space, we refer to the setting where any point of the space can be chosen as a center. This is in contrast to a *discrete* Euclidean space. Observe that for a fixed d, discrete Euclidean problems in $\mathbb{R}^d$ have bounded doubling dimension, hence covered by our framework. Furthermore, we point out that our result does not cover discrete Euclidean spaces of high dimensions, as we show in Chapter 2, in this setting, k-Center is W[1]-hard to approximate within a factor of $\sqrt{3/2} - o(1)$.

Our main contributions have two parts: (i) a new concept of metric dimension and (ii) our main technical result showing EPASes for all the aforementioned clustering problems.

Unifying Metric Spaces via Scatter Dimension

Our key conceptual contribution is a new notion of bounded metric space dimension that relaxes the standard requirement of bounded doubling dimension so that the metric spaces mentioned in Theorem 1.3.4 all "live" in a finite dimension. While the doubling dimension is often used to derive efficient algorithms for geometric problems, it proves to be too restrictive for many structured metric spaces that we consider. The ϵ-scatter dimension, on the other hand, enables us to extend parameterized approximation schemes (EPAS) beyond

traditional metric spaces like Euclidean or planar graphs. By carefully bounding the ϵ-scatter dimension of the given metric space, we are able to derive efficient EPAS algorithms that bypass coreset constructions, which are typically infeasible in high-dimensional spaces. This technique allows us to explore a wide range of clustering objectives under general norm functions, leading to a more unified and flexible framework for k-clustering problems in structured metric spaces. Our key technical result shows that two conditions—bounded ϵ-scatter dimension and monotonicity of the norm—are enough to guarantee an EPAS for clustering problems across various structured metrics.

Definition 1.3.5 (ϵ-*scatter dimension*)**.** *Given metric $M = (P, F, \delta)$, the sequence $(x_1, p_1), \ldots, (x_\ell, p_\ell) \in F \times P$ is said to be an ϵ-scattering if, whenever (x, p) appears before (x', p') in the sequence, then $\delta(x, p)$ and $\delta(x', p')$ are larger than $1 + \epsilon$ each, while $\delta(x', p) \leq 1$. The ϵ-scatter dimension of M is then defined as the length of the longest scatter, minus one.*

There are two natural interpretations. The first interpretation is as a game between two players: The *center player* who tries to claim she can cover all the points with a unit ball and the *point player* who present a counterexample. In the first round, the center player picks a center $x_1 \in F$ and the point player *refutes* the claim by presenting a point $p_1 \in P$ which is at least a factor $1 + \epsilon$ away from the (closed) unit ball around x_1, that is, $p_1 \notin \mathsf{ball}(x_1, 1 + \epsilon)$. The game continues this way: In the i-th round, the center player presents x_i such that $\{p_1, \ldots, p_{i-1}\} \subseteq \mathsf{ball}(x_i, 1)$, and the point player gives $p_i \notin \mathsf{ball}(x_i, 1 + \epsilon)$. Both players are interested in prolonging the game as much as possible. The ϵ-scatter dimension is the length of the longest possible game. In the second interpretation, one can view such sequence as a pair of ϵ-packings that are required to be sufficiently distanced: It is easy to verify (simply using triangle inequalities) that $P^* = \{p_1, p_2, \ldots, p_{\ell-1}\}$ and $F^* = \{x_2, \ldots, x_\ell\}$ are ϵ-packings of the unit (closed) balls around x_ℓ and p_1, respectively. This view immediately implies that ϵ-scatter dimension is bounded in a bounded doubling metric.

Theorem 1.3.6. *For $\epsilon \in (0, 1)$, any metric of doubling dimension d has ϵ-scatter dimension $(1/\epsilon)^{O(d)}$.*

We proceed to study the ϵ-scatter dimension of graph metrics where the set P of data points and the set F of centers are arbitrary node subsets in a graph of some fixed graph class and the distances between them are the shortest path distances.

Theorem 1.3.7. *For $\epsilon \in (0, 1)$, the ϵ-scatter dimension is $\exp\left((1/\epsilon)^{O(\mathtt{tw})}\right)$ for treewidth-$\mathtt{tw}$ graphs.*

The proof of Theorem 1.3.7 is based on a combinatorial argument that certifies treewidth lower bounds using ϵ-scattering sequences. Building on this, we present a method to reduce bounding ϵ-scatter dimension in graph classes to bounded treewidth graphs. This connection, together with the embedding result from [64], leads to the following.

Theorem 1.3.8. *For $\epsilon \in (0, 1)$, the ϵ-scatter dimension is $\exp\left(\exp(\mathit{poly}(1/\epsilon))\right)$ for planar graphs.*

Unfortunately, the bounded dimensionality does not hold in the high-dimensional (continuous) Euclidean

metric.[4] To handle the high-dimensional continuous Euclidean setting, we present a stronger version of ϵ-scatter dimension, that we call *algorithmic ϵ-scatter dimension*. The setting of the game is the same except that the center player would *optimize* to end the game early, while the point player would be interested in prolonging the game indefinitely. This means, they play against each other. A *centering strategy* is a function $\sigma\colon 2^P \to F$ that specifies how the center $x_i = \sigma(\{p_1, \ldots, p_{i-1}\})$ would be chosen by the center player, given the points $p_1, \ldots, p_{i-1}$ played in the preceding rounds. The (σ, ϵ)-scatter dimension is the maximum number of rounds when the center player always plays strategy σ, and the algorithmic ϵ-scatter dimension is the minimum (σ, ϵ)-scatter dimension over all strategies σ. We remark that our actual definition is more involved, as it considers a weighted version of the game.

Theorem 1.3.9 (Bounding Algorithmic Scatter Dimension)**.** *The continuous Euclidean space (P, F, δ), that is, $P \subseteq \mathbb{R}^d$ finite, and $F = \mathbb{R}^d$, has algorithmic ϵ-scatter dimension $O(1/\epsilon^4 \log 1/\epsilon)$.*

EPAS for General Norm Clustering: Bypassing Coresets

Now we are ready to explain our main technical result that would allow us to obtain EPAS for all metrics having bounded ϵ-scatter dimension.

A generic tool whose existence immediately implies an EPAS is an ϵ-coreset —a "compression" of an input instance (P, F, δ) into a much smaller instance so that the cost of any solution is preserved within a factor of $(1 \pm \epsilon)$. The existence of an ϵ-coreset of size depending only on ϵ and k would immediately imply an EPAS (but not vice versa): First, use the ϵ-coreset to compress the instance (P, F, δ) to (P', F', δ') where $|P'| \leq \gamma(\epsilon, k)$. Then enumerate all possible partitionings of P' into k sets $P'_1, \ldots, P'_k$ (there are at most $k^{\gamma(\epsilon,k)}$ such partitions). For each set $i \in [k]$, compute the optimal center for P'_i. We choose the partition that gives the lowest total cost.

This generic method, unfortunately, faces a serious information-theoretic limitation, that is, even for k-Center, ϵ-coresets of desirable sizes do not exist in high-dimensional Euclidean spaces [24]. Such lower bounds imply that one cannot hope to prove our (unified) results via the coreset route: While coresets are known for (k, z)-Clustering for constant z [49]—allowing to handle k-Means and k-Median in a uniform fashion—it is impossible to extend this approach to k-Center. For more complex clustering objectives, such EPASes were in fact not known even for low dimension. For example, the coreset of Braverman et al. [24] for Ordered k-Median in $\mathbb{R}^d$ has size $O_{\epsilon,d}(k^2 \log^2 n)$ and therefore does not give an EPAS even in low dimension.

Badoiu, Har-Peled, and Indyk [13] presented an EPAS for k-Center in high-dimensional Euclidean spaces (bypassing coresets in the above sense). Therefore, an obvious open question is whether their techniques can be extended to give an EPAS for any other clustering objective. Unfortunately, this is not even known for

[4] To see this, consider the sequence $(x_1, p_1) \ldots, (x_{d-1}, p_{d-1})$ where, for each $i \in [d-1]$, the point $x_i \in \mathbb{R}^d$ has i-th coordinate $1/\sqrt{2}$ and all other coordinates are zero. Define points $p_i = -x_i$ for all $i \in [d-1]$. It is easy to check that this sequence is a $(\sqrt{2}-1)$-scattering. This example implies that the ϵ-scatter dimension of continuous Euclidean metrics $\mathbb{R}^d$ can be at least $d-1$ (unbounded in ϵ). In fact, the ϵ-scatter dimension is as high as $(1/\epsilon)^{\Omega(d)}$.

simple objectives such as Priority k-Center. In fact, even the known EPASes for k-Means [94] and k-Center [13] are conceptually very different; to our knowledge, no approximation schemes handle k-Means and k-Center in a modular way.

Our main technical result is presented in the following theorem. We remark that our techniques do not rely on any coreset constructions (thus bypassing the coreset lower bounds for k-Center).

Theorem 1.3.10. *Let $\mathcal{M}$ be a class of metric spaces that is closed under scaling distances by a positive constant. There is a randomized algorithm that computes for any Norm k-Clustering instance $\mathcal{I} = (M, f, k)$ with metric $M = (P, F, \delta) \in \mathcal{M}$, and any $\epsilon \in (0, 1)$, with high probability a $(1+\epsilon)$-approximate solution if the following two conditions are met.*

(i) There is an efficient algorithm evaluating for any distance vector $\boldsymbol{x} \in \mathbb{R}^P_{\geq 0}$ the objective $f(\boldsymbol{x})$ in time $T(f)$.

(ii) There exists a function $\lambda\colon \mathbb{R}_+ \to \mathbb{R}_+$, such that for all $\epsilon > 0$, the algorithmic ϵ-scatter dimension of $\mathcal{M}$ is at most $\lambda(\epsilon)$.

The running time of the algorithm is $\exp\left(\widetilde{O}\left(\frac{k\lambda(\epsilon/10)}{\epsilon}\right)\right) \cdot \mathsf{poly}(|M|) \cdot T(f)$.

Note that the complexity of computing f appears only as a linear factor in the running time. For instance, for Socially Fair k-Median, the number m of groups affect only the computational cost of f, and therefore the running time is polynomial in m. We remark that our results extend to the setting of an approximate evaluation oracle where f can be computed to within a factor $1 \pm \epsilon$ in time $T(f)\mathsf{poly}(1/\epsilon)$ where $T(f)$ depends only on f but not on ϵ. For the sake of easier presentation we assume in this conference proceedings version that we have an exact evaluation oracle for f.

Our algorithm is clean, simple, and entirely oblivious to both the objective and the structure of the input metric.

The dependency on k in the exponent of our running time is singly exponential ($\exp(\widetilde{O}_\epsilon(k))$). In terms of k, we therefore match the running time of the fastest known EPAS for the highly restrictive special case of high-dimensional k-Means [94]. Moreover, the dependency on ϵ in the exponent could be improved by proving better bounds on the ϵ-scatter dimension of a metric space of interest, e.g., $\lambda(\epsilon) = \mathsf{poly}(1/\epsilon)$ implies the EPAS running time $\exp(\widetilde{O}(k) \cdot \mathsf{poly}(1/\epsilon))$.

1.3.3 Submodular Facility Location

As we mentioned earlier in the classical (Uncapacitated) Facility Location problem (FL), we are given a set of clients C and a set of facilities F. A feasible solution consists of an assignment where each client is assigned to a facility. The goal is to minimize the total cost, which is the sum of the distances from each client to their corresponding facility, plus the total opening cost of the facilities that serve at least one client.

In practical applications, the opening cost of a facility f often depends on the set of clients assigned to it. For instance, serving more clients may increase the facility's cost, and this relationship could be non-linear.

A natural way to handle this non-linearity is through submodularity. Specifically, we model the opening cost with a monotone submodular function $g_f : 2^C \to \mathbb{R}_+$, where $g_f(\emptyset) = 0$. The objective is to minimize the total cost, which includes both the connection costs and the facility opening costs. For each facility, the opening cost is determined by a submodular function based on the assigned clients. We refer to this as the Generalized Submodular Facility Location problem. A special case of the problem occurs when $g_f(\cdot) = g(\cdot)$ for all facilities, meaning that all facilities share the same submodular function, which we refer to as the Submodular Facility Location problem (SFL).

SFL problem is APX-hard since it includes the classical Facility Location problem (with uniform facility costs) as a special case [79]. Consequently, the best we can hope for in terms of approximation algorithms is a constant-factor approximation. However, finding such an approximation algorithm is explicitly posed as an open problem, as discussed by Svitkina and Tardos [115]. The authors present an $O(\log n)$ approximation for the the Generalized Submodular Facility Location problem, which is tight due to a reduction from the Set Cover problem by Shmoys, Swamy, and Levi [111]. The reduction from set cover does not apply for SFL problem i.e to the case where all facilities share the same submodular function $g(\cdot)$.

Svitkina and Tardos [115] also consider a special case of SFL where $g(\cdot)$ is induced by subtrees of a node-weighted tree over the clients. For this specific case, they provide a constant-factor approximation.

SFL problem models real-world situations where the cost of opening a facility is a non-linear, yet manageable, function that depends on the set of clients served by that facility. This relationship between clients and facilities can be visualized in terms of client assignments and the associated costs. Additionally, SFL is closely connected to various stochastic optimization problems that have gained significant attention in recent years [6, 75, 68, 81, 87]. These problems often involve scenarios where the connection and facility opening costs must be paid only for a random subset of activated clients, resulting in objective functions characterized by submodular opening costs. The submodular nature of these costs plays a key role in managing the complexity and variability of such optimization scenarios. An illustration is provided in Figure 1.1.

We make some progress towards the resolution of the mentioned open problem by presenting an improved approximation algorithm for SFL.

Theorem 1.3.11. *There is a polynomial-time $O(\log\log n)$-approximation algorithm for SFL.*

Our approach begins by solving a configuration LP relaxation of the problem, which can be done in polynomial time. The method proceeds in two stages: first, we sample partial assignments based on the optimal LP solution, covering a subset of clients. For the remaining clients, we reduce the connection cost using a tree embedding technique, converting the metric into a tree structure. This reduces the problem to the simpler Descendant-Leaf Assignment problem (DLA) , where an existing algorithm by Bosman and Olver [22] is adapted to achieve an $O(\log\log N)$ approximation.

The flexibility of our approach allows it to adapt to extensions of the Submodular Facility Location (SFL) problem, such as the SFL with Multiplicative Opening Costs (multSFL), where the opening cost of facility f is $g_f(S^f) = w_f \cdot g(S^f)$; and the SFL with Additive Opening Costs (addSFL), where $g_f(S^f) =$

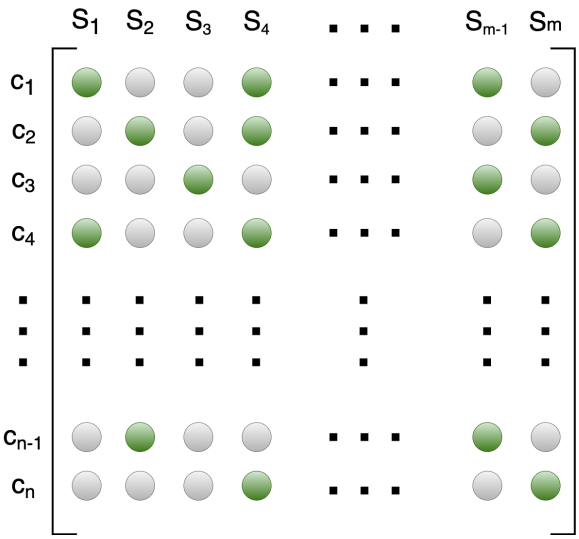


Figure 1.1: Each row represents a client, and each column represents a possible scenario. Green circles show client-to-scenario connections, indicating potential service assignments. In stochastic settings, only a random subset of clients is activated according to a probability distribution. The incurred cost consists of two components: (1) connection costs for activated clients and (2) facility opening costs. Effectively, the expected facility opening cost is a monotone submodular function of the set of clients assigned to the facility before knowing the actual scenario. The objective is to find a client-to-facility mapping that minimizes the expected total cost across all possible activation scenarios.

$p_f + g(S^f)$ for $S^f \neq \emptyset$, $g_f(\emptyset) = 0$, and $p_f \geq 0$.

Theorem 1.3.12. *There is a polynomial-time $O(\log \log n)$-approximation algorithm for* multSFL.

Theorem 1.3.13. *There is a polynomial-time $O(\log \log n)$-approximation algorithm for* addSFL.

In addition to the standard and extended variants of the Submodular Facility Location problem discussed above, our techniques can also be adapted to tackle a closely related problem known as the *Universal Stochastic Facility Location* (univFL). In this setting, we are given a set of clients C and facilities F with metric distances $d(c, f)$ as in SFL, along with an opening cost w_f for each $f \in F$. Additionally, we have oracle access to a probability distribution $\pi : 2^C \to \mathbb{R}_{\geq 0}$ specifying the probability $\pi(A)$ that a subset of clients $A \subseteq C$ is activated. A feasible solution is an *universal mapping* $\varphi : C \to F$, which assigns each client to a facility.

The cost of φ with respect to activated clients $A \subseteq C$ is given by

$$\mathsf{cost}_A(\varphi) = \sum_{c \in A} d(c, \varphi(c)) + \sum_{f \in F : \varphi^{-1}(f) \cap A \neq \emptyset} w_f,$$

which represents the connection cost for activated clients plus the opening costs for facilities that serve at least one client in A. The objective is to minimize the expected cost over all possible subsets $A \sim \pi$, i.e., $\mathbb{E}_{A \sim \pi}[\mathsf{cost}_A(\varphi)]$.

The universal nature of the solution allows for efficient, distributed decision-making in scenarios where client requests arrive dynamically over time. Let $\mathsf{OPT}\colon C \to F$ minimize $\mathbb{E}_{A\sim\pi}[\mathsf{cost}_A(\mathsf{OPT})]$, in other words OPT is an optimal (universal) mapping. We say that an algorithm for univFL is α-approximate if it returns a universal mapping φ satisfying $\mathbb{E}_{A\sim\pi}[\mathsf{cost}_A(\varphi)] \leq \alpha \cdot \mathbb{E}_{A\sim\pi}[\mathsf{cost}_A(\mathsf{OPT})]$. Interestingly, the objective function for univFL can be rewritten as:

$$\sum_{c\in C} d(c,\varphi(c)) \cdot \mathbb{P}_{A\sim\pi}[c \in A] + \sum_{f\in F} w_f \cdot \mathbb{P}_{A\sim\pi}[\varphi^{-1}(f) \cap A \neq \emptyset].$$

By interpreting $\mathbb{P}_{A\sim\pi}[R \cap A \neq \emptyset]$ for any subset $R \subseteq F$ as a monotone submodular function, we observe that univFL is nearly identical to Submodular Facility Location. This connection enables the adaptation of our techniques to achieve an improved approximation result for univFL. Notably, we exploit $\pi_{\min} := \min_{c\in C} \mathbb{P}_{A\sim\pi}[c \in A] > 0$ to ensure robustness across all possible client activations.

Theorem 1.3.14. *There is a polynomial-time $O(\log\log \frac{n}{\pi_{\min}})$-approximation algorithm for the Universal Stochastic Facility Location problem.*

The table below presents a summary of the results for these problems.

Problems	**SFL**	**multSFL**	**addSFL**	**univFL**
Known Result	$O(\log n)$ [114]			$O(\log n)$ [6]
New Result	$O(\log\log n)$	$O(\log\log n)$	$O(\log\log n)$	$O(\log\log \frac{n}{\pi_{\min}})$

Table 1.5: Summery of results for Submodular Facility Location Problem

Chapter 2

Parameterized Approximation for Socially Fair Clustering

In this Chapter, we address the following *fairness* version of the (k, z)-Clustering problem called Socially Fair (k, z)-Clustering (or equivalently Robust (k, z)-Clustering):

Socially Fair (k, z)-Clustering
Input: Instance (P, F, δ) with δ being a metric on $P \cup F$, positive integer k, a weight function $w\colon P \to \mathbb{R}_+$, and m groups $S_1, \ldots, S_m$ such that $S_i \subseteq P, P = \cup_{i\in[m]} S_i$.
Output: A k-element subset $X \subseteq F$ that minimizes $\max_{i\in[m]} \sum_{p\in S_i} w(p)\delta(p, X)^z$.

Let $n = |P|$. We remark that, in addition to generalizing k-Median and k-Means, the Socially Fair (k, z)-Clustering problem encapsulates k-Center, when each group contains a distinct singleton.

While k-Means, k-Median, and k-Center admit constant-factor approximations, it is not very surprising that Socially Fair (k, z)-Clustering is harder due to its generality: Makarychev and Vakilian [103] design a polynomial-time $O(\log m/\log\log m)$-approximation algorithm, which is tight under a plausible complexity assumption [21][1]. As this precludes the existence of efficient constant-factor approximation algorithms, recent works have focused on designing constant factor *parameterized* (FPT) approximation algorithms[2]. Along these lines, an FPT time $(3^z + \epsilon)$-approximation algorithm has been proposed and shown to be tight under the Gap Exponential-Time Hypothesis (Gap-ETH) [74]. When allowing a parameterization on the number of groups m (instead of k), Ghadiri et al. designed a $(5 + 2\sqrt{6} + \epsilon)^z$-approximation algorithm in $n^{O(m^2)}$ time [70].

Motivated by the tight lower bounds for general discrete metrics, we focus on *geometric* spaces. Geometric

[1] Note that they proved this factor for Robust k-Median, and the hardness result holds even in the line metric, unless NP $\subseteq \cap_{\delta>0}$DTIME$(2^{n^\delta})$.

[2] Throughout the chapter, parameterization refers to the natural parameter k.

spaces have a particular importance in real-world applications because data can often be represented via a (potentially large) collection of numerical attributes, that is, by vectors in a (possibly high-dimensional) geometric space. For example, in the bag-of-words model a document is represented by a vector where each coordinate specifies the frequency of a given word in that document. Such representations naturally lead to very high-dimensional data. A setting of particular interest is the high-dimensional *Euclidean space* where the metric is simply the Euclidean metric $\delta(x, y) = ||x - y||_2$.

The study of clustering problems in high-dimensional Euclidean space is an important line of research that has received significant attention in the algorithms community. It may seem intuitive to believe that it should generally (for almost any problem) be possible to algorithmically leverage the geometric structure to separate high-dimensional Euclidean from general metrics. For clustering, however, this turns out to be either false or highly non-trivial in many cases. For example, it is a long-standing open question [59] whether k-Center admits a polynomial time $(2 - \epsilon)$-approximation algorithm even in $\mathbb{R}^2$, improving the tight bound of 2 in general metrics. Interestingly enough, for the more general Euclidean k-Supplier problem, Nagarajan et al. [106] obtain an improvement over the tight bound of 3 in general metrics. The improved bounds for Euclidean k-Median and k-Means by Ahmadian et al. [7], Grandoni et al. [77], and recently by Cohen-Addad et al. [34] were breakthroughs. Concerning the more general Socially Fair (k, z)-Clustering, the tight inapproximability bound of $\Omega(\log m / \log \log m)$ in general metric continues to hold even in the line metric [21].

Similarly, the regime of FPT approximation algorithms for Euclidean clustering problems has received significant attention. Classic works design Efficient Parameterized Approximation Schemes (EPAS), that is, $(1+\epsilon)$-approximation in $f(k, \epsilon)\mathsf{poly}(n)$ time, for k-Center [13] as well as for k-Median and k-Means [94]. Recent research focuses on the design of so-called coresets [113, 48] whose existence implies an EPAS if their size only depends on k and the error parameter ϵ.

In the real space $\mathbb{R}^d$, it is important to distinguish between the *discrete* and the *continuous* settings. In the discrete setting, both the point set P and the candidate center set F are finite subsets of $\mathbb{R}^d$ while in the continuous setting, centers can be chosen anywhere in the metric space, that is, $F = \mathbb{R}^d$. A separate line of research has studied the contrast between continuous and discrete versions. For example, while discrete clustering variants are clearly polynomial-time solvable for constant k by trivial enumeration, the continuous versions of k-Center and k-Median are known to be NP-hard even for $k = 2$ [56] in high-dimensional Euclidean space. Also in terms of polynomial-time approximability, stronger lower bounds were shown by Cohen-Addad et al. [43] for the continuous versions. Indeed, there have been systematic research efforts in understanding these geometric clustering problems [46, 42, 43].

2.1 Overview of Techniques

Improved FPT Approximation in High-Dimensional Discrete Euclidean Space.

Our algorithm underlying Theorem 1.3.1 is a slight modification of the factor-$(3^z + \epsilon)$ FPT approximation algorithm for general metrics by Goyal and Jaiswal [74]. Our main technical contribution lies in the improved analysis. A key component of the analysis by Goyal and Jaiswal is a simple projection property of metric spaces (see Lemma 2.1.1 below). We argue that under minor additional assumptions, this property can be strengthened in Euclidean space. The resulting *assigment lemma* (see Lemma 2.2.1) is at the heart of our analysis and its proof relies on several new ideas and technically involved ingredients.

We briefly review the algorithm by Goyal and Jaiswal [74]. Their algorithm consists of two main steps. First, they compute a (κ, λ)-bicriteria solution $B \subseteq F$, that is, the cost of B is bounded by $\kappa\mathsf{OPT}$ and the cardinality of B is bounded by λk. Specifically, they obtain guarantees $\kappa = 1 + \epsilon$ and $\lambda = O\left(\log^2 n/\epsilon^2\right)$ for sufficiently small $\epsilon > 0$. In the second step, they extract a feasible solution from the (infeasible) bi-criteria solution B by enumerating all k-subsets of B and outputting the one of minimum cost.

Their analysis is based on proving the existence of a k-subset of B whose cost is at most $(3^{z-1}(\kappa + 2))\mathsf{OPT}$, which can be bounded by $(3^z + \epsilon)\mathsf{OPT}$ assuming z being constant. Since the algorithm enumerates all k-subsets, this provides an upper bound on the cost of the algorithm. The key component of their existential argument is the following simple property of metric spaces, which we call *projection lemma*. It is convenient to think of O as an optimal solution and B as a bicriteria solution with $|B| > |O|$ but the lemma holds for any sets B, O.

Lemma 2.1.1 (Projection Lemma)**.** *Let (Y, δ) be a metric space, and $B \subseteq Y$. Then for any set $O \subseteq Y$, there exists an assignment $\sigma\colon O \to B$ such that, for all $o \in O$ and $y \in Y$, we have*

$$\delta(y, \sigma(o)) \leq 2\delta(y, o) + \delta(y, B)\,. \tag{2.1}$$

Intuitively, their lemma allows them to "project" the optimal solution O onto a k-subset $\sigma(O) \subseteq B$ of the bicriteria solution so that for any client $y \in Y$, the distance $\delta(y, \sigma(O))$ can be charged to $\delta(y, O)$ and $\delta(y, B)$. If fact, the number 3 in the approximation factor $3^z + \epsilon$ corresponds to the sum $(2 + 1)$ of the coefficients in front of $\delta(y, o)$ and $\delta(y, B)$.

In this chapter, we study the setting where Y is a discrete Euclidean metric (P, F, δ), that is, where P, F are finite subsets of $\mathbb{R}^d$ and δ is the Euclidean distance. A natural attempt to improve the approximation factor in the Euclidean setting is to reduce the coefficients in front of the terms $\delta(y, o)$ and $\delta(y, B)$ in the projection lemma. Unfortunately, this straightforward approach fails: The projection lemma is tight even on the line metric; see Figure 2.1.

It turns out that slightly enlarging the projection space is already sufficient to bypass this obstacle. More

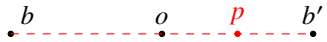


Figure 2.1: This example shows that the projection lemma is tight even for the 1-dimensional Euclidean space. Let $o = 0$ be the optimum facility located at the origin and serving client $p = 1/2$. Let $b' = 1$ be the facility in B that serves p and let $b = \sigma(o) = -1$ be the facility in B nearest to o. We have $\mathsf{OPT} = 1/2$, which also equals the cost of B. However $\delta(p, \sigma(o)) = 3/2 = 2 \times \delta(p, o) + 1 \times \delta(p, b')$. Combining multiple such examples in orthogonal directions and sharing facility b shows that the approximation ratio of the algorithm of Goyal and Jaiswal [74] approaches 3 in the discrete Euclidean space.

specifically, we project onto the *midpoint closure*

$$\mathsf{cl}(B) = B \cup \left\{ \pi_F\left(\frac{b+b'}{2}\right) : b, b' \in B \right\}, \tag{2.2}$$

of the bicriteria solution where $\pi_F(p)$ represents the closest facility in F to point p. This step exploits that the metric space is embedded into $\mathbb{R}^d$ (so that the midpoints exist).

While on the algorithmic side a slight modification of the original algorithm is sufficient for the improvement, the analysis requires several new ideas and technically involved ingredients. To prove a strengthened version of the projection lemma (called assignment lemma) we set up a factor-revealing geometric optimization problem in the plane; see (2.3) in Definition 2.2.2 below. We call the optimum objective γ_β of this problem *displacement ratio*. Roughly speaking, this ratio corresponds to the maximum ratio between the left-hand and the right-hand side of (2.1) in Lemma 2.1.1. However, we project to $\mathsf{cl}(B)$ rather than B and impose some additional minor restrictions. By a careful and technically involved analysis of this optimization problem we can upper bound the displacement ratio in the Euclidean setting by $1 - \epsilon_0$ for some universal constant $\epsilon_0 > 0$ as long as two obstructions are avoided. The first obstruction occurs in any configuration similar to the one in Figure 2.1 above where the bi-criteria solution contains two facilities b, b' so that o is near to the mid-point of b and b'. However, in such a configuration facility o certifies that $b'' = \pi_F((b + b')/2)$ must be close to o allowing us to assign o to b'' contained in the mid-point closure. The second obstruction arises if p is *β-near*, that is, within a small distance β from o (but there is no facility in B such as b' as in the first obstruction). For β approaching 0, the displacement ratio of β-near points can approach 1 even if when projecting to the mid-point closure of B. To account for β-near points, we therefore cannot resort to the assignment lemma. However, the overall contribution of β-near points to the cost of the projected solution can be shown to be very small. More details of the algorithm and its analysis are provided in Section 2.2.1.

Hardness of Discrete k-Center

Our proof constructs an instance of the discrete k-Center from an instance of Multi-Colored Independent Set problem, which is known to be $W[1]$-hard. In Multi-Colored Independent Set, we are given a k-partite graph G with a k-partition of the vertices $V_1, \ldots, V_k$, and the goal is to determine if there is an independent set that contains precisely one node from each set $V_i, i \in [k]$. The gadget in our construction is a set of nearly equidistant binary code words. Such code words with relative Hamming distance roughly $1/2$ and logarithmic

length are known to exist (see Ta-Shma [116]). The high level idea is as follows. We associate each vertex of G with a unique code word of suitable length t. Then, we generate a data point in P for each vertex and edge of G by using code word(s) associated with the corresponding vertices. The construction guarantees the following crucial properties: (i) The Hamming distance between the data points of vertices is roughly t. (ii) The Hamming distance between a data point of vertex $v \in V_i$ and a data point of an edge e is roughly t if e is incident on $V_i \setminus \{v\}$ and is roughly $3t/2$ otherwise. (iii) The Hamming distance between the data points of edges is at least (close to) $3t/2$. Thus, the construction forces us to pick data points of vertices as centers in our solution and guarantees that the optimum cost of the k-Center instance is roughly t if and only if there is an independent set in G. As a result, approximating the cost of the k-Center instance better than a (roughly) $(3/2)^{1/q}$ factor would imply $W[1] = \mathsf{FPT}$. That is because the cost of a k-Center instance is the maximum ℓ_q distance between a data point and its closest selected center, and hence, approximating this cost better than the mentioned factor allows us to distinguish between Yes and No cases of an arbitrary instance of Multi-Colored Independent Set.

Approximation Scheme for Metrics of Sub-Logarithmic Doubling Dimension.

Our algorithm comprises two main components, both based on standard techniques from the literature: instance compression and decomposition of the doubling metric into smaller balls. However, it becomes evident that a natural construction based on these standard techniques for Socially Fair (k, z)-Clustering faces serious information-theoretic limitations, as explained below. One natural idea for compressing a Socially Fair (k, z)-Clustering instance is to reduce the number of groups, as each group can be further compressed using a (k, z)-Clustering coreset (such coresets exist [48]). This reduction yields a significantly smaller instance. If we could reduce the number of groups to $m' \ll m$ while approximately preserving the cost for every solution, we could obtain an EPAS as follows. First, apply a (k, z)-Clustering coreset to every group of the compressed instance to obtain another Socially Fair (k, z)-Clustering instance with m' groups, each containing $g(k, \epsilon)$ points, where g is some function that represents the size of (k, z)-Clustering coreset. It is essential to note that this compression is acceptable for obtaining an EPAS since the coreset of a group approximately preserves the (k, z)-Clustering cost of the group. Next, enumerate all k-partitions of the points within each group to find potential solutions. Finally, return the solution that has the minimum Socially Fair (k, z)-Clustering cost. Unfortunately, because Socially Fair (k, z)-Clustering captures k-Center (and consequently faces a coreset lower bound of $2^{\Omega(d)}$ in Euclidean space of dimension d [24]), the number of new groups must satisfy $m' \geq 2^{\Omega(d)}$. Consequently, the running time of this algorithm is $k^{2^{\Omega(d)}}\mathsf{poly}(n, m)$, which is doubly exponential in d. It is worth noting that this algorithm matches the running time of [3] and does not yield an EPAS for sub-logarithmic dimension.

Furthermore, if we explore an alternative approach and utilize the coreset of k-Center, it is not immediately clear how to extend the coreset of k-Center to reduce the number of groups in an instance of Socially Fair (k, z)-Clustering. This is because, firstly, we would require a mapping between the old groups and the new groups, and secondly, this mapping should ideally approximately preserve the Socially

Fair (k, z)-Clustering cost for every solution.

Another potential method for compressing the instance involves reducing the number of points in set P, rather than altering the groups, with the hope of designing an EPAS that can exploit the smaller P (without concern for the number of groups). However, for this approach to succeed, it is essential to establish a bijection between the old and new groups. Yet, it remains uncertain whether such a bijection exists. In typical coreset constructions, each point in the coreset P' of P has a weight that is the sum of the weights of the points in its local neighborhood in P which it is supposed to represent in P'. However, these points in P could potentially belong to different groups, making it challenging to establish the mapping between groups.

The core idea of our approach is to work with an alternative and more general definition of groups that permits a point to participate in different groups with varying weights. In this revised definition, instead of viewing groups as subsets of points, we treat each group as a weight function that assigns non-negative real values to points. This flexibility allows different weights to be assigned to the same point by different groups, which can, in fact, be of practical interest. Utilizing this new definition, we can devise an approach for compressing the points such that each point in the compressed instance can have a weight for group g that represents the sum of the weights of nearby points in g that were filtered out during compression. Essentially, this enables us to approximately preserve the group costs. With this approach and additional technical work that leverages the standard ball decomposition technique for doubling metrics, we derive a coreset for Socially Fair (k, z)-Clustering that can be employed to construct an EPAS for doubling metrics with sub-logarithmic dimension.

2.2 High-Dimensional Discrete Euclidean Space

2.2.1 FPT Approximation Algorithm for Socially Fair (k, z)-Clustering

In this section, we exploit non-trivial properties of the Euclidean metric to prove the following result that breaches the barrier of 3^z-approximation for Socially Fair (k, z)-Clustering in general metrics.

Theorem 1.3.1 (High-Dimensional Euclidean Space)**.** *There exists a universal constant $\eta_0 > 0.0006$ such that for any constant positive integer z, there is a factor $3^z(1-\eta_0)$ FPT approximation algorithm for Socially Fair (k, z)-Clustering in discrete Euclidean space $\mathbb{R}^d$ that runs in time $2^{O(k \log k)}\mathsf{poly}(m, n, d)$.*

Recall from Section 2.1 that our approach begins with computing a (κ, λ)-bicriteria solution B to the Socially Fair (k, z)-Clustering instance employing the algorithm proposed by Goyal-Jaiswal [74]. As we argued, it is sufficient to prove the existence of a k-subset of B whose cost is within a constant factor of optimal. The result by Goyal and Jaiswal [74] is based on the following simple projection lemma for general metrics whose proof we state here for the sake of later reference.

Lemma 2.1.1 (Projection Lemma)**.** *Let (Y, δ) be a metric space, and $B \subseteq Y$. Then for any set $O \subseteq Y$, there*

exists an assignment $\sigma : O \to B$ *such that, for all* $o \in O$ *and* $y \in Y$*, we have*

$$\delta(y, \sigma(o)) \leq 2\delta(y, o) + \delta(y, B) . \tag{2.1}$$

Proof. For each $o \in O$, define $\sigma(o)$ as $\pi_B(o)$, the point in B closest in distance to o. Notice that for any $o \in O$, $y \in Y$, we have $\delta(y, \sigma(o)) \leq \delta(y, o) + \delta(o, \sigma(o))$ by triangle inequality. The lemma follows by combining this with $\delta(o, \sigma(o)) = \delta(o, B) \leq \delta(o, \pi_B(y)) \leq \delta(y, o) + \delta(y, B)$. □

This lemma itself is tight even in 1-dimensional Euclidean space (as we showed in Figure 2.1). In order to get around this issue, we make use of the property of our geometric space. Given the instance (P, F, δ) embedded into the Euclidean space and the bicriteria solution B, we project to the mid-point closure $\mathsf{cl}(B)$ as defined in (2.2).

Notice that $|\mathsf{cl}(B)| = O\left(|B|^2\right)$. Let O be the optimal solution. For $\beta > 0$ we say that client $p \in P$ is *β-far* (from O w.r.t. B) if $\delta(p, O) \geq \beta \cdot \delta(p, B)$, and we say that client p is *β-near* otherwise. The key of our analysis is the following strengthening of the projection lemma for Euclidean space, which we call assignment lemma.

Lemma 2.2.1 (Assignment Lemma)**.** *Let* $\beta_0 = 0.05$ *and let* $B \subseteq \mathbb{R}^d$*. Then, for any* $O \subseteq \mathbb{R}^d$*, there exists an assignment* $\sigma : O \to \mathsf{cl}(B)$ *such that, for all* β_0*-far points* $p \in \mathbb{R}^d$*, we have* $\delta(p, \sigma(O)) \leq (1 - \epsilon_0)(2\delta(p, O) + \delta(p, B))$ *where* $\epsilon_0 > 0.002$*.*

To prove Lemma 2.2.1, we start with defining the assignment function σ. Take any facility $o \in O$ and let $b = \pi_B(o)$. We assume w.l.o.g. that the instance is rotated so that p, b and o lie in the plane spanned by the first two coordinates. For the sake of easier notation, we identify p, b, o by points in $\mathbb{R}^2$. Further, by translation and scaling, we assume that o coincides with the origin and that $b = (-1, 0)$. Let $q = (0, 1)$ be the mirror image of b. Let α be a parameter to be fixed (we later set it to 0.6). We define $\sigma(o)$ based on the position of o relative to an α-ball. Specifically, $\sigma(o) = b$ if the α-ball centered at a point q contains no facility from B; otherwise, $\sigma(o)$ is the projection $\pi_{\mathrm{cl}(B)}(o)$ of o onto the mid-point closure of B.

Our goal is to analyze the displacement of a client p under the assignment rule σ. Recall from the proof of Lemma 2.1.1 that if $\sigma(o)$ is simply the projection onto B, then a client p, when served by facilities o and b' in sets O and B respectively, incurs a cost of at most $2||p - o|| + ||p - b'||$. We wish to show that the assignment cost in our algorithm is strictly smaller than this upper bound (under certain assumptions). We prove this by bounding the ratio of these two quantities.

Definition 2.2.2 (Displacement Ratio)**.** *For a given small constant* $\beta > 0$*, let the* displacement ratio *be defined as*

$$\gamma_\beta = \max_{\substack{p \in \mathbb{R}^d \setminus \mathit{ball}(o,\beta), \\ b' \in \mathbb{R}^d \setminus \mathit{ball}(o,1)}} \left\{ \frac{||p - \sigma(o)||}{2||p - o|| + ||p - b'||} \right\} . \tag{2.3}$$

Let S be the plane spanned by b, p, and o. After the appropriate rotations and translations we mentioned earlier, S would coincide with the x-y plane. In what follows, we also restrict b' to lie in plane S as well.

This follows due to Claim 2.2.3 mentioned below. For cleaner analysis, we defer the proof of this claim to Section 2.2.1.

Claim 2.2.3. *The maximum displacement ratio γ_β is achieved by some b' that lies in plane S containing points b, o, and p.*

To show the lemma, we demonstrate that γ_β can be upper-bounded by $1 - f(\alpha, \beta)$ for some $f(\alpha, \beta) > 0$, where $f(\cdot)$ is a function dependent on α, β and the geometry of O and B.

Claim 2.2.4. *Following the assignment rule σ, we have that $\gamma_\beta \leq 1 - f(\alpha, \beta)$.*

For the sake of notation, we drop the subscript of γ_β everywhere in the proof.

Proof. Given O and B, we consider a facility $o \in O$. Let $b \in B$ the closest facility to o, and $b' \in B$ the bicriteria solution that serves client p. Now consider the α-ball around q and β-ball around o, for the ease of analyse we consider half plane above x-coordinate the same arguments hold for half plane below x-coordinate. We distinguish two cases.

Case 1: $\mathsf{ball}(q, \alpha) \cap B \neq \emptyset$. Suppose that B contains a facility b'' lying inside the α-ball around q. Given that $b \in B$ is the closest facility to o, it follows that $b'' \in (\mathsf{ball}(q, \alpha) \backslash \mathsf{ball}(o, 1))$. In this case $\sigma(o) = \pi_{\mathrm{cl}(B)}(o)$. Hence $\sigma(o)$ is no farther from o than the facility $\pi_F((b + b'')/2)$ nearest to the midpoint of b and b''. Notice that the optimal center o certifies the existence of a point in F nearby the midpoint of b and b''. The point $o \in F$ shows that $\pi_F((b + b'')/2)$ has distance at most α to o because $||\pi_F((b + b'')/2) - (b + b'')/2|| \leq ||(b + b'')/2 - o||$. Hence we obtain $||\sigma(o) - o|| \leq \alpha$, therefore $||p - \sigma(o)|| \leq ||p - o|| + \alpha$ (see Figure 2.2). Recall that the aim is to upper bound the displacement ratio 2.2.2 for client p, notice that $||p - o|| + ||p - b'|| \geq 1$, we obtain:

$$\begin{aligned} \gamma = \frac{||\sigma(o) - p||}{2||p - o|| + ||p - b'||} &\leq \frac{||p - o|| + \alpha}{||p - o|| + 1} \\ &\leq 1 - \frac{1 - \alpha}{||p - o|| + 1} \\ &\leq 1 - \frac{1 - \alpha}{2} \end{aligned}$$

Case 2: $\mathsf{ball}(q, \alpha) \cap B = \emptyset$. in the second case, where the α-ball does not contain a facility from B, we argue that the points o, $\sigma(o) = b$, and b' are far enough from a co-linear position. This allows us to argue that the triangle inequality in the proof of Lemma 2.1.1 is not tight. Towards this, we divide the space into four regions that could contain client p, as indicated in Figure 2.3.

We define q_1 and q_2 as two points of intersection between $\mathsf{ball}(o, 1)$ and $\mathsf{ball}(q, \alpha)$. See Figure 2.4 for an illustration. We assume that p lies the half plane above the x-axis. (The case where p lies below the x-axis is symmetric.) Now, consider q_3 as the midpoint of q and q_1. Furthermore, define the region H as the area above

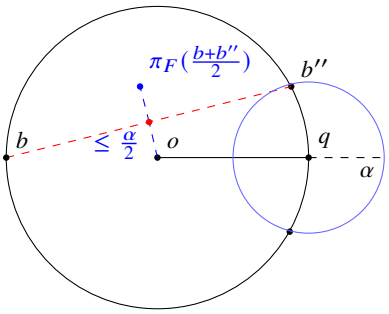


Figure 2.2: The midpoint of b and b'' is shown by red dot, $||(b+b'')/2 - o|| \leq \frac{\alpha}{2}$ and thus $||\sigma(o) - o|| \leq \alpha$.

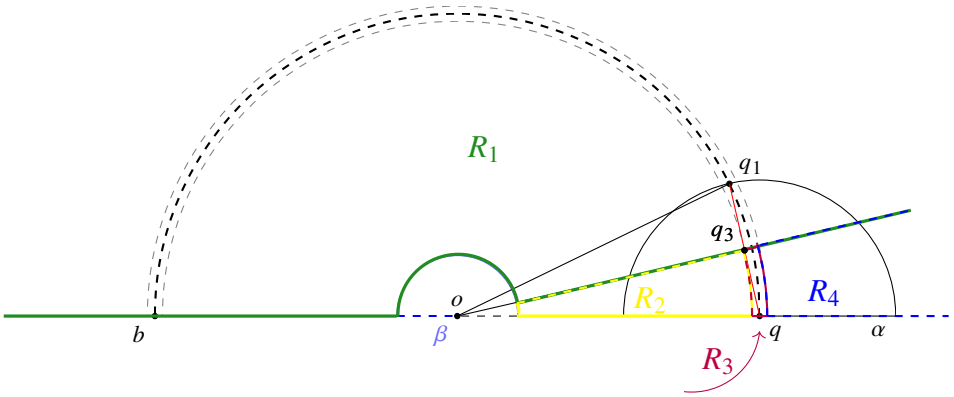


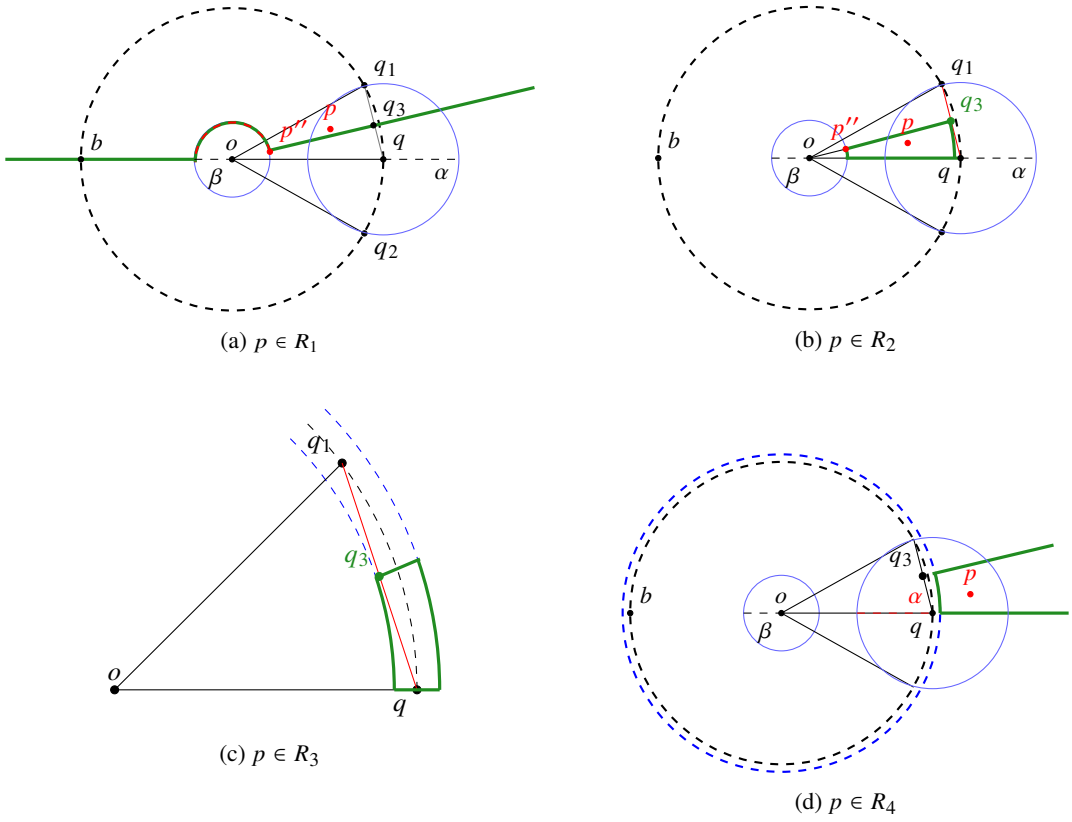
Figure 2.3: The dashed black circle depicts $\mathsf{ball}(o,1)$, while the dashed gray circles represent $\mathsf{ball}(o,1-\omega)$ and $\mathsf{ball}(o,1+\omega)$. Regions R_1, R_2, R_3, and R_4 are outlined with green, yellow, purple, and blue borders respectively.

the lines passing through(q_3, o) and (o, b). We define region $R_1 = H \backslash \mathsf{ball}(o, \beta)$. Next, consider $(1-\omega)$ and $(1+\omega)$ balls around o, H' is defined as the area below the line passing through (o, q_3) and above the line passing through (o, q), we define $R_2 = (\mathsf{ball}(o, 1-\omega) \backslash \mathsf{ball}(o, \beta)) \cap H'$, $R_3 = (\mathsf{ball}(o, 1+\omega) \backslash \mathsf{ball}(o, 1-\omega)) \cap H'$, and $R_4 = H' \backslash \mathsf{ball}(o, 1+\omega)$, the regions are indicated in Figure 2.3.

- $p \in R_1$, Let b'' be the closest point to p not in the interior of $\mathsf{ball}(o,1)$, and let p' be the point on the boundary of $\mathsf{ball}(o,\beta)$ that is closet to p. Let p'' be the point where the segment (o, q_3) intersects the boundary of $\mathsf{ball}(o,\beta)$, that is, $p'' = (\beta\cos\theta, \beta\sin\theta)$ where $\theta = \angle q_3 o q_1$. Notice that $\cos\theta = 1 - \frac{\alpha^2}{4}$, see figure 2.4a for an illustration. First, we assume p is inside $\mathsf{ball}(o, 1+2\beta)$ in the region of R_1.

  Observation 2.2.5. *For any $\epsilon_1, \epsilon_2, X, Y \geq 0$:*

$$\frac{X - \epsilon_1 + Y}{X + Y} \leq \frac{X - \epsilon_1 + Y + \epsilon_2}{X + Y + \epsilon_2}$$



(a) $p \in R_1$ (b) $p \in R_2$

(c) $p \in R_3$ (d) $p \in R_4$

Figure 2.4: $o \in O$ is an optimum solution, $b \in B$ is the closest bicriteria solution to o, 1-ball around o is shown as a dashed circle, α-ball around q and β-ball around o are shown in blue, $(1-\omega)$ and $(1+\omega)$ around o are shown as blue dashed circles. The regions are specified by green borders.

Consider assigning p via p' to b. We bound the displacement cost as follows:

$$\begin{aligned}
\gamma_\beta = \frac{||\sigma(o) - p||}{2||p-o|| + ||p-b'||} &\leq \frac{||b-p'|| + ||p-p'||}{2||p'-o|| + ||p'-b''|| + ||p-p'||} \\
&\leq \frac{||b-p''|| + ||p-p'||}{2\beta + 1 - \beta + ||p-p'||} \\
&\leq \frac{\sqrt{(\beta\cos(\theta)+1)^2 + (\beta\sin(\theta))^2} + ||p-p'||}{1+\beta+||p-p'||} \\
&= \frac{\sqrt{\beta^2 + 2\beta\cos\theta + 1} + ||p-p'||}{1+\beta+||p-p'||} \\
&= \frac{\sqrt{(1+\beta)^2 - \frac{\beta\alpha^2}{2}} + ||p-p'||}{1+\beta+||p-p'||}
\end{aligned}$$

We assume $||p - p'|| \leq 1 + \beta$, and by observation 2.2.5, we obtain:

$$\gamma_\beta \leq \frac{(1+\beta)(1+\sqrt{(1-\frac{\beta\alpha^2}{2})})}{2(1+\beta)} \leq \frac{1}{2} + \frac{\sqrt{1^2 - \frac{2\beta\alpha^2}{4} + \frac{\beta^2\alpha^4}{16}}}{2} \leq \frac{1}{2} + \frac{1 - \frac{\beta\alpha^2}{4}}{2}$$
$$= 1 - \frac{4 + \beta\alpha^2}{8}$$

Second, let's assume that the client p is distant from o and positioned within region R_1 outside $\mathsf{ball}(o, 1 + 2\beta)$, we can bound γ_β as follows:

$$\gamma_\beta \leq \frac{1 + ||o - p||}{2||o - p||} \leq \frac{1 + 1 + 2\beta}{2(1 + 2\beta)} = \frac{1 + \beta}{1 + 2\beta} = 1 - \frac{\beta}{1 + 2\beta}$$

- $p \in R_2$, we obtain the best location for b' is when it lies on the point q_1 (see Figure 2.4b for an illustration), the cost of displacement is as follows:

$$\gamma = \frac{||\sigma(o) - p||}{2||p - o|| + ||p - b'||} \leq \frac{||p - o|| + 1}{||p - o|| + ||p' - o|| + ||p' - p|| + ||p - b'||}$$
$$\leq \frac{||p - o|| + 1}{||p - o|| + ||p'' - o|| + ||p'' - b'||}.$$

To calculate $||p'' - b'||$ we can consider rotated p'' and b' so that $p'' = (\beta, 0)$ and $b' = (\cos\theta, \sin\theta)$, then:

$$\gamma \leq \frac{||p - o|| + 1}{||p - o|| + \beta + \sqrt{(\cos(\theta) - \beta)^2 + (\sin(\theta))^2}}$$

Assume $\beta \leq \frac{\alpha}{12}$ then we have $||p - o|| \leq 1 - \beta$, hence:

$$\gamma \leq \frac{2 - \beta}{1 + \sqrt{\beta^2 - 2\beta(1 - \frac{\alpha^2}{4}) + 1}}$$
$$\leq \frac{2 - \beta}{1 + (1 - \beta)\sqrt{1 + 2\beta\frac{\alpha^2}{4}}}$$
$$= 1 - \frac{\sqrt{1 + \beta\frac{\alpha^2}{2}}(1 - \beta) - (1 - \beta)}{1 + (1 - \beta)\sqrt{1 + \beta\frac{\alpha^2}{2}}}$$

- $p \in R_3$, we claim the distance from p to the closest bicriteria solution b' can be bounded as $||p - b'|| \geq \frac{\alpha}{3}$. Suppose $\alpha \leq \frac{6}{10}$, we define $z = 1 - ||q_3 - o|| \leq \frac{\alpha}{12}$, consider p''' the closest point to p on the line (q_3, q), hence $||p - p'''|| \leq \frac{\alpha}{6}$, now we have $\frac{\alpha}{2} \leq ||p''' - b'|| \leq ||p''' - p|| + ||p - b'||$, then we obtain

$||p - b'|| \geq \frac{\alpha}{3}$.

$$\begin{aligned}\gamma = \frac{||\sigma(o) - p||}{2||p - o|| + ||p - b'||} &= \frac{||p - o|| + 1}{||p - o|| + (1 - \frac{\alpha}{12}) + \frac{\alpha}{3}} \\ &= \frac{||p - o|| + 1}{||p - o|| + 1 + \frac{\alpha}{4}} \\ &\leq \frac{2}{2 + \frac{\alpha}{4}} \\ &= 1 - \frac{\alpha}{8 + \alpha}\end{aligned}$$

- $p \in R_4$, assume p and b' are in the same location, therefore:

$$\gamma = \frac{||\sigma(o) - p||}{2||p - o|| + ||p - b'||} \leq 1 - \frac{||p - o|| - ||b - o||}{2||p - o||}$$

Notice that $||p - o|| \geq 1 + \frac{\alpha}{12}$, consider two cases either $||p - o|| \leq 1 + \alpha$ or $||p - o|| > 1 + \alpha$:

  - $||p - o|| \leq 1 + \alpha$

$$\gamma \leq 1 - \frac{1 + \frac{\alpha}{12} - 1}{2(1 + \alpha)} = 1 - \frac{\alpha}{24(1 + \alpha)}$$

  - $||p - o|| > 1 + \alpha$

$$\begin{aligned}\gamma \leq 1 - \frac{||p - o|| - 1}{2||p - o||} &\leq \frac{1}{2} + \frac{1}{2(1 + \alpha)} \\ &= 1 - \frac{1 + \alpha}{2(1 + \alpha)}\end{aligned}$$

Therefore, by examining the position of p in the regions, we establish that γ_β is upper-bounded by $1 - f(\alpha, \beta)$. Consequently, Lemma 2.2.1 is substantiated by showing the existence of an $\alpha_0 \leq 0.6$ and a sufficiently small $\beta_0 \leq 0.05$ such that $\gamma_{\beta_0} \leq 1 - f(\alpha_0, \beta_0) = 1 - \epsilon_0 \leq 0.9978$. □

In the proof of Theorem 1.3.1, we show that this new assignment property is enough to derive an improved FPT approximation for Socially Fair (k, z)-Clustering in Euclidean space. Since the assignment σ maps every facility in O uniquely to a facility in $\mathsf{cl}(B)$, this implies that $\sigma(O)$ is a feasible solution of cost at most $(3^z \cdot (1 - \eta_0))\mathsf{OPT}$. This certifies the existence of a feasible solution being a subset of $\mathsf{cl}(B)$ with the desired approximation factor. Hence, we can find such a solution in FPT time by enumeration.

Now we are ready to prove Theorem 1.3.1 (restated for convenience).

Theorem 1.3.1 (High-Dimensional Euclidean Space)**.** *There exists a universal constant $\eta_0 > 0.0006$ such that for any constant positive integer z, there is a factor $3^z(1 - \eta_0)$ FPT approximation algorithm for Socially Fair (k, z)-Clustering in discrete Euclidean space $\mathbb{R}^d$ that runs in time $2^{O(k \log k)}\mathsf{poly}(m, n, d)$.*

Proof. Let $B \subseteq \mathbb{R}^d$ denote a $(1 + \epsilon_0, k \cdot \ln n^2/\epsilon_0^2)$ bi-criteria solution by applying the algorithm of [74]. We denote the total cost of the set of β_0-near (β_0-far) points by $\mathsf{BIC}_\mathrm{n}, \mathsf{OPT}_\mathrm{n}, \mathsf{ALG}_\mathrm{n}$ ($\mathsf{BIC}_\mathrm{f}, \mathsf{OPT}_\mathrm{f}, \mathsf{ALG}_\mathrm{f}$) in a

bi-criteria solution, an optimum solution, and the solution returned by our algorithm, respectively. Note that from the definition of β_0-near points and the (clustering) cost, we have that $\mathsf{OPT}_\mathrm{n} \le \beta_0 \cdot \mathsf{BIC}_\mathrm{n} \le \beta_0 \cdot \mathsf{OPT}$. Consequently, for the set of β_0-far points, we get $\mathsf{OPT}_\mathrm{f} \ge (1-\beta_0) \cdot \mathsf{OPT}$.

For the set of β_0-near points, the best bound on the cost that our algorithm can achieve is that of Lemma 2.1.1. Therefore, we get $\mathsf{ALG}_\mathrm{n} \le (2\mathsf{OPT}_\mathrm{n} + \mathsf{BIC}_\mathrm{n})$. On the other hand, by Lemma 2.2.1, we do save some cost on the β_0-far points as we have $\mathsf{ALG}_\mathrm{f} \le \big((1-\epsilon_0)(2\mathsf{OPT}_\mathrm{f} + \mathsf{BIC}_\mathrm{f})\big)$. Moreover $\mathsf{BIC}_\mathrm{n} + \mathsf{BIC}_\mathrm{f} = \mathsf{BIC} \le (1+\epsilon_0)\mathsf{OPT}$ by the choice of the bicriteria solution. Putting these factors together, we obtain

$$
\begin{aligned}
\mathsf{ALG}_\mathrm{n} + \mathsf{ALG}_\mathrm{f} &\le 3^{z-1} \cdot \left[(2\mathsf{OPT}_\mathrm{n} + \mathsf{BIC}_\mathrm{n}) + (1-\epsilon_0)^z (2\mathsf{OPT}_\mathrm{f} + \mathsf{BIC}_\mathrm{f})\right] \\
&\le 3^{z-1} \cdot \left[2\beta_0^z \mathsf{OPT} + 2(1-\epsilon_0)^z (1-\beta_0^z)\mathsf{OPT} + \mathsf{BIC}_\mathrm{n} + (1-\epsilon_0)^z \mathsf{BIC}_\mathrm{f}\right] \\
&\le 3^{z-1} \cdot \left[2\beta_0^z \mathsf{OPT} + 2(1-\epsilon_0)^z (1-\beta_0^z)\mathsf{OPT} + \mathsf{BIC}_\mathrm{n} + \mathsf{BIC}_\mathrm{f}\right] \\
&\le 3^{z-1} \cdot \left[2\big(\beta_0^z + (1-\epsilon_0)^z (1-\beta_0^z)\big)\mathsf{OPT} + (1+\epsilon_0)\mathsf{OPT}\right] \\
&\le 3^{z-1} \cdot \left[\big(2\beta_0^z + 2(1-\epsilon_0)^z - 2(1-\epsilon_0)^z \beta_0^z + 1 + \epsilon_0\big)\mathsf{OPT}\right] \\
&\le 3^{z-1} \cdot \left[\big(2(1-\epsilon_0)^z + 2\beta_0^z \epsilon_0 z + 1 + \epsilon_0\big)\mathsf{OPT}\right] \quad \text{(using Bernoulli's inequality[3])} \\
&\le 3^{z-1} \cdot \Big[\underbrace{\big(2(1-\epsilon_0)^z + (2\beta_0^z z + 1)\epsilon_0 + 1\big)}_{3-\epsilon_1}\mathsf{OPT}\Big]
\end{aligned}
$$

where in the first inequality, we used the approximate triangle inequality $(a+b+c)^z \le 3^{z-1} \cdot (a^z + b^z + c^z)$ and in the second inequality, we use the fact that $\mathsf{OPT}_\mathrm{n} + \mathsf{OPT}_\mathrm{f} = \mathsf{OPT}$ and thus, the maximizer for $2\mathsf{OPT}_\mathrm{n} + 2(1-\epsilon_0)^z \mathsf{OPT}_\mathrm{f}$ occurs at the maximum of OPT_n. Next, to bound the above under-braced expression, we introduce a new parameter ϵ_1, which we prove below in claim 2.2.6 that $\epsilon_1 \ge 0.0018$. Finally, we have

$$
\begin{aligned}
\mathsf{ALG}_\mathrm{n} + \mathsf{ALG}_\mathrm{f} &\le 3^{z-1} \cdot (3-\epsilon_1)\mathsf{OPT} \\
&\le 3^z \cdot (1-\eta_0)\mathsf{OPT} \text{ where } \eta_0 = \frac{\epsilon_1}{3}
\end{aligned}
$$

Claim 2.2.6. *For any integer z, we have that $2(1-\epsilon_0)^z + \epsilon_0(1 + 2\beta_0^z z) \le 2 - \epsilon_1$*

Proof. Let $2(1-\epsilon_0)^z + \epsilon_0(1+2\beta_0^z z) \le 1.9982$. Note that, since from Lemma 2.2.1 we have that $\beta_0 = 0.05$ and $\epsilon_0 > 0.002$, we get lower bound for ϵ_1 as follows (the left-hand side is maximized at $z = 1$): $\epsilon_1 > 2 - 2(1-\epsilon_0) - \epsilon_0(1+2\beta_0) > 0.0018$. □

It remains to analyze the running time of the algorithm. In the initial phase of our algorithm, we invoke the Goyal-Jaiswal bi-criteria algorithm [74]. Subsequently, we evaluate all possible k-subsets of $\mathsf{cl}(\mathrm{B})$, whose number is bounded by $\binom{(\lambda \cdot k)^2}{k} \le (e\lambda)^{2k}$. This leads to an overall running time $O\left((e\lambda)^{2k} \cdot nk\right)$ where

$\lambda = O\left(\frac{k}{\epsilon_0^2} \cdot \ln^2 n\right)$, therefore we have $O\left((e\frac{k}{\epsilon_0^2} \cdot \ln^2 n)^{2k} \cdot nk\right) = (\frac{k}{\epsilon_0})^{O(k)} \cdot n^{O(1)}$. Let $\eta_1 = 3^z \cdot \eta_0$, Then for $\eta_0 > 0.0006$ and constant z, ϵ_1 simplifies to $\frac{\eta_1}{3^{z-1}}$, resulting in an overall complexity of $(k/\epsilon_1)^{O(k)} \cdot n^{O(1)}$, which is FPT in terms of k.

□

Proof of Claim 2.2.3

We first define the ratio γ'_β:

$$\gamma'_\beta = \max_{\substack{p \in \mathbb{R}^d \setminus \mathsf{ball}(o,\beta), \\ b' \in S \setminus \mathsf{ball}(o,1)}} \left\{ \frac{||p - \sigma(o)||}{2||p - o|| + ||p - b'||} \right\}.$$

Note that for the sake of the optimization problem, b and o are fixed points and therefore, the plane S is a function of the location of point p. Hence, we write the plane S as S_p. We prove that $\gamma_\beta \leq \gamma'_\beta$. Let b' and p be the points optimizing γ_β. Suppose b' is not on S_p. We start by choosing an orthonormal basis (x, y, z) for the linear space spanning b, o, p and b', and fixing a system of coordinates. Towards this, let $z = b - o$, $x = \frac{p - \hat{p}}{||p - \hat{p}||}$, where $\hat{p}$ is the orthogonal projection of p on the line containing b and o. Then fix $y = \frac{b' - \hat{b'}}{||b' - \hat{b'}||}$, where $\hat{b'}$ is the orthogonal projection of b' on S, the plane containing b, o, and p. Now we fix the origin of the coordinate system to be in the center of the following disk [4].

Let D be the disk the perimeter of which is defined as the circle $\partial(\mathsf{ball}(o, 1)) \cap \partial(\mathsf{ball}(q, \alpha))$, where ∂ indicates the boundary of the closed space. Notice that after this translation of the coordinate system, D is contained in the x-y plane and that q, p, and b' can be represented as $q = (0, 0, q_z)$, $p = (p_x, 0, p_z)$, and $b' = (b'_x, b'_y, b'_z)$.

For any point $s = (s_x, s_y, s_z)$ let $\bar{s} = (s_x, s_y, 0)$ denote its projection onto the x-y plane. See Figure 2.5 for an illustration of the projections $\bar{q}$, $\bar{p}$, and $\bar{b'}$ of q, p, and b' on the x-y plane, respectively.

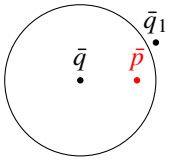


Figure 2.5: An illustration of D and the projection points of p, q and b'.

Proposition 2.2.1. *If $b'_y \neq 0$ then there exists $b'' \notin$ interior(ball$(o, 1)$) so that $||b'' - p||^2 < ||b' - p||^2$ and so that $b'' \in$ interior(ball(q, α)) if and only if $b' \in$ interior(ball(q, α)).*

Proof. Consider b'' as a rotation of b' with respect to the line (o, q), i.e., a rotation that preserves disk D. We choose to rotate by the angle that will place $\overline{b''}$ as close to $\overline{p}$ as possible, i.e., $\overline{b''}$ will be co-linear with $\overline{q}$ and $\overline{p}$ ($\overline{b''}_y = \overline{q}_y = \overline{p}_y = 0$). Note that the rotation preserves the distances $||b'' - o||$ and $||b'' - q||$ to

[4] As a clarification, we temporarily relocate the origin of the coordinate system to the center of the disk and we fixed $q = (0, 0, q_z)$.

be equal to $||b' - o||$ and $||b' - q||$. To see that, assume that $\check{b'}$ is the orthogonal projection of b' on the line containing o and q We have that $||b' - o|| = \sqrt{(||b' - \check{b}||)^2 + (||\check{b'} - o||)^2}$. The claim follows since the orthogonal projection of b'' on the line would also land on $\check{b'}$. The argument for $||b'' - q||$ follows in a similar way. As a consequence, since b' is outside $\mathsf{ball}(o, 1)$, then b'' is also outside $\mathsf{ball}(o, 1)$. Also, b'' will fall outside of $\mathsf{ball}(q, \beta)$ if and only if b' is outside $\mathsf{ball}(q, \beta)$.

We observe that $||b'_z - p_z||^2$ is constant w.r.t. rotations. This allows us to reduce our argument to analyzing the change of squared distances within the x–y plane.

$$\begin{aligned}
||b' - p||^2 &= ||b'_x - p_x||^2 + ||b'_y - p_y||^2 + ||b'_z - p_z||^2 \\
&= ||\overline{b'} - \overline{p}||^2 + ||b'_z - p_z||^2 \\
&> ||\overline{b''} - \overline{p}||^2 + ||b'_z - p_z||^2 \\
&= ||b''_x - p_x||^2 + ||b''_y - p_y||^2 + ||b'_z - p_z||^2 \\
&= ||b''_x - p_x||^2 + ||b''_y - p_y||^2 + ||b''_z - p_z||^2 \\
&= ||b'' - p||^2.
\end{aligned}$$

It remains to observe that $||b'' - p||^2 < ||b' - p||^2$ implies $||b'' - p|| < ||b' - p||$.

□

By Proposition 2.2.1, we obtain that $\overline{b'}$ must be co-linear with $\overline{p}$ and $\overline{q}$, therefore $\gamma_\beta \leq \gamma'_\beta$ and point b' may be assumed to be on the same plane S as points b, o, and p. The fact that $b'' \notin \mathsf{interior}(\mathsf{ball}(o, 1))$ ensures b remains the closest point in F to o when replacing b' with b''. The property that $b'' \in \mathsf{interior}(\mathsf{ball}(q, \alpha))$ if and only if $b' \in \mathsf{interior}(\mathsf{ball}(q, \alpha))$ ensures that, after replacing b' with b'', o is assigned to the midpoints of b'' and b if and only if we it was assigned to the midpoint of b' and b before the replacement. The two properties together guarantee that the replacement of b' with b'' does not change $\sigma(o)$.

2.3 Hardness of Discrete k-Center

For this section, we use the following explicit construction of the so-called *η-balanced error-correcting codes* from a recent result of Ta-Shma [116] which we rephrase for our purposes as follows:

Theorem 2.3.1. *Let $\eta \in (0, 1/2)$ be a positive constant. Then there is an algorithm that computes, for any given number $s \in \mathbb{N}$, an s-element set $B \subseteq \{0,1\}^t$ of binary vectors of dimension $t = O(\log s/\eta^{2+o(1)})$ such that for any $b \in B$, its Hamming weight $||b||_1$ and for any $b' \in B \setminus \{b\}$, the Hamming distance $||b - b'||_1$ both lie in the interval $[(1/2 - \eta)t, (1/2 + \eta)t]$. The running time of the algorithm is $O(st)$.*

Proof. Ta-Shma [116] gives an explicit construction of a $t \times \lceil \log_2 s \rceil$ binary matrix generating a linear, binary, error-correcting code of message length $\lceil \log_2 s \rceil$, block length $t = O(\log s/\eta^{2+o(1)})$, and pairwise Hamming

distance between $(1/2-\eta)t$ and $(1/2+\eta)t$. Since the code is linear, it contains the zero code word. Hence each code word has Hamming weight in $[(1/2-\eta)t, (1/2+\eta)t]$. The time for constructing the matrix is polynomial in $\log s$ and t. Using the generating matrix, at least s many non-zero code words can be enumerated in time $O(st)$, which dominates the time for computing the matrix.

□

We leverage balanced error correcting codes as gadget in our hardness proof for discrete k-Center. For any binary vector $b \in \{0,1\}^t$, we denote by $\bar{b}$ the binary vector obtained by flipping each coordinate in b.

Theorem 1.3.2 (Hardness in Discrete Euclidean Space)**.** *For any constant positive integer q and any positive constant $\eta > 0$, there exists a function $d(k,n) = O(k \log n)$ such that there is no factor-$(3/2-\eta)^{1/q}$ FPT approximation algorithm for the discrete k-Center problem in $\mathbb{R}^{d(k,n)}$ under the ℓ_q metric unless $W[1] = FPT$. Moreover, for the ℓ_2 metric this hardness holds even for some dimension $O(\log n)$, that is, independently of k.*

Proof. We show a reduction from Multi-Colored Independent Set, which is known to be $W[1]$-hard [50]. The input is a k-partite graph $G = (V, E)$ with k-partition $V_1, \ldots, V_k$. The question is if there is an independent set that is *multi-colored*, that is, it has precisely one node from each set V_i, $i \in [k]$. W.l.o.g. we assume that each V_i contains at least one node that is adjacent to all nodes $V \setminus V_i$. Adding such nodes, we can additionally assume that $|V_i| = n/k$ for each $i \in [k]$ where $n = |V|$.

Fix some constant $\eta \in (0, 1/2)$. Using Theorem 2.3.1, we construct a set $B \subseteq \{0,1\}^t$ of n nearly equidistant code words of dimension $t = O(\log n/\eta^{2+o(1)})$. We map each node $u \in V$ uniquely to some non-zero code word $b(u) \in B$. We construct a k-Center instance in $\mathbb{R}^{k \cdot t}$ as follows. We subdivide the coordinates of each point in $\mathbb{R}^{k \cdot t}$ into k *blocks* each containing t consecutive coordinates. In our set P of data points, we introduce for each node $v_i \in V_i$, $i \in [k]$, the point $p(v_i) \in P$ in which the ith block equals $b(v_i)$ and all other coordinates are zero. For each edge $(v_i, v_j) \in E$, $v_i \in V_i$, $v_j \in V_j$ for distinct $i, j \in [k]$ we create a point $p(v_i, v_j) \in P$ in which the ith block equals $\overline{b(v_i)}$, the jth block equals $\overline{b(v_j)}$, and all other coordinates are zero. No further points are added to P. We set the number of centers to be k completing the construction of the k-Center instance.

Let $i \in [k]$ and $v_i, v_i' \in V_i$ be distinct vertices. We have that $||p(v_i) - p(v_i')||_q^q \leq ||b(v_i) - b(v_i')||_1 \leq (1/2 + \eta)t$ by Theorem 2.3.1. Let $v_j \in V_j$, $j \in [k]$ such that $(v_i, v_j) \in E$. By Theorem 2.3.1, we have that

$$\begin{aligned} ||p(v_i') - p(v_i, v_j)||_q^q &\leq ||b(v_i') - \overline{b(v_i)}||_1 + ||\overline{b(v_j)}||_1 \\ &\leq (t - ||b(v_i') - b(v_i)||_1) + (t - (1/2 - \eta)t) \\ &\leq (t - (1/2 - \eta)t) + (1/2 + \eta)t \\ &\leq (1 + 2\eta)t\,. \end{aligned}$$

Hence if there is a multi-colored independent set I for G then $X = \{\, p(u) \mid u \in I \,\}$ is a k-element set such that $\delta(p, X)^q \leq (1 + 2\eta)t$ for any $p \in P$ under the ℓ_q metric, which gives an upper bound of $(1 + 2\eta)t$ on the k-Center objective in the completeness case.

For analyzing the soundness case, assume that there is no multi-colored independent set for G. Consider an arbitrary k-element set $X \subseteq V$. We say that $x \in X$ *covers* $p \in P$ if $\delta(p,x)^q < (3/2-3\eta)t$. We claim that there is some $p \in P$ not covered by any center in X. The correctness of this claim implies that any parameterized approximation algorithm with approximation ratio strictly better than $((3/2-3\eta)/(1+2\eta))^{1/q}$ implies that $W[1] = \mathsf{FPT}$ and thus the theorem.

In order to prove this claim, we assume for the sake of contradiction, that all $p \in P$ are covered by some center in X. First, we argue that w.l.o.g. X contains no point of the form $p(v_i, v_j)$ where $(v_i, v_j) \in E$. In fact, for any $g \notin \{i, j\}$, we have that

$$\begin{aligned} ||p(v'_g) - p(v_i, v_j)||_q^q &\geq ||b(v'_g)||_1 + ||\overline{b(v_i)}||_1 + ||\overline{b(v_j)}||_1 \\ &\geq (1/2-\eta)t + 2(t - (1/2+\eta)t) \\ &= (3/2-3\eta)t\,. \end{aligned} \tag{2.4}$$

Hence $p(v_i, v_j)$ can cover $p(v'_g)$ only if $g = i$ or $g = j$. Similarly, $p(v_i, v_j)$ can cover $p(v'_g, v'_h)$ only if $i = g$ and $j = h$. But then these points would be covered by $p(v_i)$ as well and hence we could replace $p(v_i, v_j)$ with $p(v_i)$. We therefore assume that X contains only points of the form $p(v_i)$.

We claim that X is multi-colored. Otherwise, there would be some V_i that contains no point from X. By our initial assumption, V_i contains some point v_i that is adjacent to all points $V \setminus V_i$. Assuming $k \geq 3$ there exists at least one V_j, $j \neq i$ that contains at most one node from X. If V_j intersects X then let $v_j \in V_j \cap X$, and otherwise let v_j be an arbitrary node in V_j. By our assumption $(v_i, v_j) \in E$. If $v_j \in X$ then

$$\begin{aligned} ||p(v_j, v_i) - p(v_j)||_q^q &\geq ||\overline{b(v_j)} - b(v_j)||_1 + ||\overline{b(v_i)}||_1 \\ &\geq t + (t - (1/2+\eta)t) \\ &= (3/2-\eta)t \end{aligned} \tag{2.5}$$

as the jth block of $p(v_j)$ equals $b(v_j)$ and the ith block of $p(v_j, v_i)$ equals $\overline{b(v_j)}$. If $v_j \notin X$ then for any $v_h \in X$ we have $h \notin \{i, j\}$. Thus $||p(v_i, v_j) - p(v_h)||_q^q \geq (3/2-3\eta)t$, which follows as in (2.4). Hence $p(v_i, v_j)$ would not be covered showing that X is multi-colored. Since X is multi-colored it can not be an independent set. Hence there exists some edge (v_i, v_j) such that $v_i, v_j \in X$ but then $||p(v_i) - p(v_i, v_j)||_q^q \geq (3/2-\eta)t$, $||p(v_j) - p(v_i, v_j)||_q^q \geq (3/2-\eta)t$, and $||p(v_h) - p(v_i, v_j)||_q^q \geq (3/2-3\eta)t$ for any $v_h \in X$, $h \notin \{i, j\}$, which follows as in (2.5) and (2.4), respectively. Hence $\delta(p(v_i, v_j), X) \geq (3/2-3\eta)t$, implies that $p(v_i, v_j)$ is not covered.

We complete the proof by noting that the dimension of the instance can be reduced to $O(\log n)$ for Euclidean metrics by using the Johnson-Lindenstrauss transform with sufficiently small (constant) error parameter. □

2.4 EPAS for Metrics of Sub-Logarithmic Doubling Dimension

In this section, we show an EPAS for Socially Fair (k,z)-Clustering in metrics of sub-logarithmic doubling dimension. This result complements the hardness result of Section 2.2 (Theorem 1.3.2). Towards our goal, we prove the following result.

Theorem 1.3.3 (EPAS for Doubling Metric of Sub-Logarithmic Dimension)**.** *There is an algorithm that computes $(1+\epsilon)$-approximate solution, for every $\epsilon > 0$, for Socially Fair (k,z)-Clustering in the metric of doubling dimension d in time $f(k,d,\epsilon,z)\mathsf{poly}(m,n)$, where $f(k,d,\epsilon,z) = \left(\left(\frac{2^z}{\epsilon}\right)^d k\log k\right)^{O(k)}$.*

Note that the above algorithm runs in FPT time for $d = o(\log n)$. We also remark that the above result can be extended to the continuous $\mathbb{R}^d$. Throughout this section, we assume that the weight aspect ratio $\frac{\max_{p\in P} w(p)}{\min_{p'\in P} w(p')}$ and the distance aspect ratio $\frac{\max_{p,p'\in P}\delta(p,p')}{\min_{p\neq p'\in P}\delta(p,p')}$ are bounded by $\mathsf{poly}(n)$, some polynomial in n. For $p\in P$ and any number $r\geq 0$, denote by $\mathsf{ball}(p,r)$ to be the closed ball centered at p of radius r. We prove the theorem in two steps: first, in Section 2.4.1 we show an algorithm to obtain a coreset for the problem, and then, in Section 2.4.2 we show how to use this coreset to get the algorithm of Theorem 1.3.3.

2.4.1 Coreset for Socially Fair (k,z)-Clustering

The key idea for constructing coresets for Socially Fair (k,z)-Clustering crucially relies on the following alternate but equivalent definition of the problem. In this definition, we are given $\mathcal{I} = (F, P\subset\mathcal{M}, \mathcal{W})$, where either $F=\mathcal{M}$ or $F\subseteq\mathcal{M}$, where $\mathcal{M}$ is doubling metric of dimension d, defined by the metric function δ. A *group* is a weight vector $\mathbf{w}\in\mathcal{W}$ such that $\mathbf{w}: P\to\mathbb{R}_{\geq 0}$. Given $X\subseteq F$, the distance vector $\boldsymbol{\delta}_P(X)$ is defined as $\boldsymbol{\delta}_P(X)[p] = \delta(p,X)^z$, for each $p\in P$. The cost of X for a group $\mathbf{w}\in\mathcal{W}$ is defined as $c(\mathbf{w},X) = \mathbf{w}\cdot\boldsymbol{\delta}_P(X)$. For a Socially Fair (k,z)-Clustering instance $\mathcal{I} = (F,P,\mathcal{W})$, the cost of X is defined as $\mathsf{cost}(\mathcal{I},X) = \max_{\mathbf{w}\in\mathcal{W}}\mathsf{cost}(\mathbf{w},X)$. The cost of the instance $\mathcal{I} = (F,P,\mathcal{W})$ is

$$\mathsf{OPT}(\mathcal{I}) = \min_{X\subseteq F,|X|=k}\max_{\mathbf{w}\in\mathcal{W}}\mathsf{cost}(\mathbf{w},X)$$

Whenever the instance $\mathcal{I}$ is clear from context, we will just write OPT. Notice that, in the original Socially Fair (k,z)-Clustering, a group is given by $S\subseteq P$, and this can be captured by weight vector $\mathbf{w}[p]=0$ for $p\notin S$ and $w(p)$ otherwise. We prove the following coreset exists for Socially Fair (k,z)-Clustering.

Theorem 2.4.1 (Coreset for Socially Fair (k,z)-Clustering)**.** *Given an instance $\mathcal{I} = (F,P,\mathcal{W})$ of Socially Fair (k,z)-Clustering in doubling metric of dimension d and $0<\epsilon\leq 1$, there is an algorithm that, in time $\left(\frac{2^z}{\epsilon}\right)^{O(d)}\mathsf{poly}(n,m)$, computes another instance $\mathcal{I}' = (F,P',\mathcal{W}')$ of Socially Fair (k,z)-Clustering with $P'\subseteq P : |P'| = \left(\frac{2^z}{\epsilon}\right)^{O(d)} kz\log n$ such that for any $X\subseteq F$ with $|X|=k$,*

$$(1-\epsilon)\mathit{cost}(\mathcal{I},X)\leq\mathit{cost}(\mathcal{I}',X)\leq(1+\epsilon)\mathit{cost}(\mathcal{I},X).$$

We remark that the above theorem yields a coreset of clients, and not of groups, and hence, the total size of coreset is comparable to the original instance. However, we will show later that such coreset is sufficient to get a parameterized approximation scheme with parameters k and d. We would also like to point out that the exponential dependency on d on the point set size of the coreset is inevitable since Socially Fair (k,z)-Clustering captures k-Center, for which such a lower bound is known [24, 16]. To see that our notion of coreset for Socially Fair (k,z)-Clustering coincides with the regular notion of coreset for k-Center, note that in this setting each group contains a single distinct point.

In the next section, we describe the algorithm of Theorem 2.4.1.

The Algorithm. Our algorithm (See Algorithm 1) is inspired by the grid construction approach of [84] that yields coresets for k-Median and k-Means. Given an instance $\mathcal{I} = (F, P, \mathcal{W})$ of Socially Fair (k,z)-Clustering, the first step is to start with an (α,β)-bicriteria solution $B = \{b_i\}_{i\in[\beta k]}$ that opens at most βk facilities with the guarantee that $\mathsf{cost}(\mathcal{I}, B) \leq \alpha \cdot \mathsf{OPT}$, for some constants $\alpha, \beta \geq 1$. Let $R = \sqrt[z]{\frac{\mathsf{cost}(\mathcal{I},B)}{\alpha\tau}}$, where $\tau := \max_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w}||_1$. Let $\Delta = \frac{\max_{p\in P,\mathbf{w}\in\mathcal{W}} \mathbf{w}[p]}{\min_{p\in P,\mathbf{w}\in\mathcal{W}} \mathbf{w}[p]}$ be the weight aspect ratio of $\mathcal{I}$. Then, for each $b_i \in B$, consider the balls $\mathcal{B}_i^j := \mathsf{ball}(b_i, 2^j R)$, for $j \in \{0, \cdots, \lceil 2\log(\alpha n\Delta)\rceil\}$. Note that, for $\mathbf{w} \in \mathcal{W}$ and $p \in P$ with $\mathbf{w}[p] > 0$, it holds that $\delta(p, B) \leq R\sqrt[z]{\alpha n\tau}$, since $\delta(p,B) \leq \sqrt[z]{\frac{\mathsf{cost}(\mathcal{I},B)}{\mathbf{w}[p]}} \leq \sqrt[z]{\frac{\alpha\tau}{\mathbf{w}[p]}} R \leq R\sqrt[z]{\alpha n\Delta}$. Hence, we have that every point $p \in P$ is contained in some ball $\mathcal{B}_i^j$. For $b_i \in B$, let $\mathcal{Q}_i^j = \mathcal{B}_i^j - \mathcal{B}_i^{j-1}$, for $j = \{1, \cdots, \lceil 2\log(\alpha\Delta)\rceil\}$, be the ring between $\mathcal{B}_i^j$ and $\mathcal{B}_i^{j-1}$, with $\mathcal{Q}_i^0 = \mathcal{B}_i^0$. Decompose every ball $\mathcal{B}_i^j$ into smaller balls each of radius $\frac{\epsilon}{40\alpha} R 2^j$ using the fact that the metric is a doubling metric. These balls can intersect, so we assign every point $p \in P$ to exactly one ball (for example, by associating p to the smallest ball containing p, breaking ties arbitrarily).
For every ball $\mathcal{B}_i^j$ and every smaller ball t of $\mathcal{B}_i^j$ with $|t \cap \mathcal{Q}_i^j| \neq \emptyset$, pick an arbitrary point $p' \in t \cap \mathcal{Q}_i^j$ as the *representative* of (the points in) $t \cap \mathcal{Q}_i^j$, and add p' to the coreset P' with group weight vectors as follows. Corresponding to every group vector $\mathbf{w} \in \mathcal{W}$, create a new group vector $\mathbf{w}' \in \mathcal{W}'$. Then,$\mathbf{w}'[p'] := \sum_{p\in t\cap\mathcal{Q}_i^j} \mathbf{w}(p)$. Intuitively, $\mathbf{w}'[p']$ captures the total weight of points of $\mathbf{w}$ in $t \cap \mathcal{Q}_i^j$. This concludes the coreset construction.

The high-level idea above is to decompose each ball $\mathcal{B}_i^j$ into smaller balls and pick a distinct point as the representative of points in the non-empty decomposed ball. Additionally, such representative p' participates in the group $\mathbf{w}'$ with weight which is sum of the weights of points in $\mathbf{w}$ that are represented by p'. However, we want to decompose the ball $\mathcal{B}_i^j$ into smaller balls in a way that the total number of balls remains the same, irrespective of the radius of the ball. This is necessary as for higher values of j, this number would depend on n, if we are not careful. While this does not seem to help much, as the radius of the decomposed balls is much large for higher values j, it actually does the trick: since the points in these balls are far from b_i, and hence their connection cost to b_i is also large. This allows us to represent the radii of larger balls in terms of the connection cost of its points to B, thus bounding the error in terms of the cost of B, which in turn is bounded by $\alpha\mathsf{OPT}$, which gives us the desired guarantee.

Algorithm 1: Coreset construction for Socially Fair (k, z)-Clustering

Data: Instance $\mathcal{I} = (F, P, \mathcal{W})$ of Socially Fair (k, z)-Clustering, (α, β)-bicriteria solution B for $\mathcal{I}$

Result: Coreset $\mathcal{I}' = (F, P', \mathcal{W}')$ for $\mathcal{I}$

Let $P' \leftarrow \emptyset$;
Let $\mathbf{w}' \leftarrow \mathbf{0}$ for $\mathbf{w}' \in \mathcal{W}'$;
Let $\tau \leftarrow \max_{\mathbf{w} \in \mathcal{W}} ||\mathbf{w}||_1$;
Let $R = \sqrt[z]{\frac{\mathsf{cost}(\mathcal{I}, B)}{\alpha \tau}}$;
For each $b_i \in B$ for $j \in \{0, 1, \cdots, \lceil 2 \log \alpha n \Delta \rceil\}$, let $\mathcal{B}_i^j = \mathsf{ball}(b_i, 2^j R)$, and let $Q_i^j = \mathcal{B}_i^j - \mathcal{B}_i^{j-1}$ with $Q_i^0 = \mathcal{B}_i^0$;
Decompose each ball $\mathcal{B}_i^j$ into balls of radius each $\frac{\epsilon}{\alpha 3^{z+2}} 2^j R$; `// e.g., use Lemma 2.4.4`
Associate each point $p \in P$ to a smallest ball containing p breaking ties arbitrary;
foreach $i \in [k]$ **do**
  foreach $j \in \{0, 1, \cdots, \lceil 2 \log \alpha n \Delta \rceil\}$ **do**
    foreach *smaller ball* t *of* $\mathcal{B}_i^j$ **do**
      if $\exists p \in t \cap Q_i^j$ **then**
        $P' \leftarrow P' \cup p$;
        foreach $w \in \mathcal{W}$ **do**
          Set the corresponding weight vector $\boldsymbol{w}'[p] = \sum_{p' \in t \cap Q_i^j} \boldsymbol{w}[p']$;
        end
        break;
      end
    end
  end
end
return $\mathcal{I}' := (F, P', \mathcal{W}')$;

Analysis. First, let us bound $|P'|$. Note $|P'| = O(|B| \log(\alpha n) \left(\frac{\alpha 3^z}{\epsilon}\right)^{O(d)})$, assuming $\Delta = \mathsf{poly}(n, m)$. We will use the following bi-criteria algorithm for Socially Fair (k, z)-Clustering due [103].

Theorem 2.4.2 ([103]). *There exists a polynomial time algorithm for Socially Fair (k, z)-Clustering that, for every $\gamma \in (0, 1)$, outputs at most $\frac{k}{1-\gamma}$ centers whose cost is bounded by $\frac{e^{O(z)}}{1-\gamma}$ times the optimal cost.*

Invoking Algorithm 1 with the above bi-criteria solution for $\gamma = 1/2$, we get $|P'| = \left(\frac{2^z}{\epsilon}\right)^{O(d)} kz \log n$, as desired. Note that the running time of the algorithm is $\left(\frac{2^z}{\epsilon}\right)^{O(d)} \mathsf{poly}(n)$.

We now argue that $\mathcal{I}'$ is indeed a coreset for $\mathcal{I}$. Let $\mathsf{OPT}(\mathcal{I})$ be the cost of optimal solution for $\mathcal{I}$. Fix a feasible solution $X \subseteq F, |X| = k$ and let $\hat{\boldsymbol{w}} \in \mathcal{W}$ be a maximizer of the Robust k-Median cost of $\mathcal{I}$ for X. We claim that, for any $\boldsymbol{w}' \in \mathcal{W}'$, it holds that

$$\mathsf{cost}(\boldsymbol{w}, X) - \epsilon \mathsf{cost}(\hat{\boldsymbol{w}}, X) \leq \mathsf{cost}(\boldsymbol{w}', X) \leq (1 + \epsilon) \mathsf{cost}(\hat{\boldsymbol{w}}, X),$$

where $\boldsymbol{w} \in \mathcal{W}$ is the corresponding weight vector to $\boldsymbol{w}'$. Fix any $\boldsymbol{w}' \in \mathcal{W}'$ and the corresponding $\boldsymbol{w} \in \mathcal{W}$.

For $p \in P$, let $r(p) \in P'$ be the representative of p. Using the inequality[5] $|a^z - b^z| \leq |(a-b)(a+b)^{z-1}|$, for $a, b \geq 0$, we have that the total error $\mathcal{E} := |\mathsf{cost}(\mathbf{w}', X) - \mathsf{cost}(\mathbf{w}, X)|$ is bounded by,

$$\mathcal{E} \leq \sum_{p \in P} |(\mathbf{w}[p]\delta(p,X)^z - \mathbf{w}[p]\delta(r(p),X)^z)| \leq \sum_{p \in P} \mathbf{w}[p] |(\delta(p,X) - \delta(r(p),X))(\delta(p,X) + \delta(r(p),X))^{z-1}|$$

Note that $|(\delta(p,X) - \delta(r(p),X))| \leq |(\delta(p,r(p)) + \delta(r(p),X) - \delta(r(p),X))| \leq \delta(p,r(p))$. Further, $\delta(p,X) + \delta(r(p),X) \leq 2\delta(p,X) + \delta(p,r(p))$. Hence,

$$\mathcal{E} \leq \sum_{p \in P} \mathbf{w}[p] \cdot \delta(p,r(p))(2\delta(p,X) + \delta(p,r(p)))^{z-1}.$$

To bound $\mathcal{E}$, we divide the points in P in three parts, and bound the errors on each part separately. Let $P_R := \{p \in P \mid \delta(p,B) \leq R \;\&\; \delta(p,X) \leq R\}$, $P_B := \{p \in P \mid \delta(p,B) > R \text{ and } \delta(p,X) \leq \delta(p,B)\}$, and $P_X := \{p \in P \mid \delta(p,X) > R \text{ and } \delta(p,B) \leq \delta(p,X)\}$.

For $p \in P_R$, we have $\delta(p,r(p)) \leq \frac{\epsilon R}{\alpha 3^{z+1}}$, and $\delta(p,X) \leq R$, and hence we have,

$$\begin{aligned}
\mathcal{E}_R &\leq \sum_{p \in P_R} \mathbf{w}[p] \frac{\epsilon R}{\alpha 3^{z+1}} \left(2R + \frac{\epsilon R}{\alpha 3^{z+1}}\right)^{z-1} \\
&\leq \frac{\epsilon R^2}{\alpha 3^{z+1}} \left(2 + \frac{\epsilon}{\alpha 3^{z+1}}\right)^{z-1} \sum_{p \in P_R} \mathbf{w}[p] \\
&\leq \frac{\epsilon \mathsf{OPT}}{9\alpha\tau} \sum_{p \in P_R} \mathbf{w}[p] \\
&< \frac{\epsilon}{3}\mathsf{OPT} \qquad \text{since } \tau \geq ||\mathbf{w}||_1.
\end{aligned}$$

For $p \in P_B$, suppose $2^j R \geq \delta(p,B) > 2^{j-1}R$, for some $j \geq 1$. Then, note that $\delta(p,r(p)) \leq 2\frac{\epsilon 2^j R}{\alpha 3^{z+2}} < 2\frac{2\epsilon\,\delta(p,B)}{\alpha 3^{z+2}} < \frac{\epsilon}{\alpha 3^z}\delta(p,B)$. Hence, we bound $\mathcal{E}_B$ using the fact $\delta(p,X) \leq \delta(p,B)$,

$$\mathcal{E}_B \leq \sum_{p \in P_B} \mathbf{w}[p] \frac{\epsilon}{\alpha 3^z} \delta(p,B)^z \left(2 + \frac{\epsilon}{\alpha 3^z}\right)^{z-1} \leq \frac{\epsilon}{3\alpha} \sum_{p \in P_B} \mathbf{w}[p]\delta(p,B)^z \leq \frac{\epsilon}{3}\mathsf{OPT}(\mathcal{I}).$$

For $p \in P_X$, suppose $\delta(p,B) > R$, then $2^j R \geq \delta(p,B) > 2^{j-1}R$, for some $j \geq 1$. In this case, we have $\delta(p,r(p)) \leq 2\frac{\epsilon 2^j R}{\alpha 3^{z+2}} < 2\frac{2\epsilon\,\delta(p,B)}{\alpha 3^{z+2}} < \frac{\epsilon}{\alpha 3^z}\delta(p,B) \leq \frac{\epsilon}{\alpha 3^z}\delta(p,X)$. Otherwise, $\delta(p,B) \leq R$, in which case $\delta(p,r(p)) \leq 2\frac{\epsilon R}{\alpha 3^{z+2}} < \frac{2\epsilon}{\alpha 3^{z+2}}\delta(p,X)$. Hence,

$$\mathcal{E}_X \leq \sum_{p \in P_X} \mathbf{w}[p] \frac{\epsilon}{\alpha 3^z} \delta(p,X)^z \left(2 + \frac{\epsilon}{\alpha 3^z}\right)^{z-1} \leq \frac{\epsilon}{3\alpha} \sum_{p \in P_X} \mathbf{w}[p]\delta(p,X)^z \leq \frac{\epsilon}{3}\mathsf{cost}(\mathbf{w}, X).$$

[5] This can be proved using induction on z.

Now,

$$\begin{aligned}\mathsf{cost}(\boldsymbol{w}', X) &\leq \mathsf{cost}(\boldsymbol{w}, X) + \epsilon\mathsf{OPT}(\mathcal{I}) \\ &\leq \mathsf{cost}(\boldsymbol{w}, X) + \epsilon\mathsf{cost}(\hat{\boldsymbol{w}}, X) \qquad \text{since } \mathsf{OPT}(\mathcal{I}) \leq \mathsf{cost}(\mathcal{I}, X) = \mathsf{cost}(\hat{\boldsymbol{w}}, X) \\ &\leq (1+\epsilon)\mathsf{cost}(\hat{\boldsymbol{w}}, X) \qquad \text{since } \mathsf{cost}(\boldsymbol{w}, X) \leq \mathsf{cost}(\hat{\boldsymbol{w}}, X).\end{aligned}$$

Similarly, $\mathsf{cost}(\boldsymbol{w}', X) \geq \mathsf{cost}(\boldsymbol{w}, X) - \epsilon\mathsf{OPT}(\mathcal{I}) \geq \mathsf{cost}(\boldsymbol{w}, X) - \epsilon\mathsf{cost}(\hat{\boldsymbol{w}}, X)$ since $\mathsf{OPT}(\mathcal{I}) \leq \mathsf{cost}(\hat{\boldsymbol{w}}, X)$.

Now, we finish the proof as follows.

$$\mathsf{cost}(\mathcal{I}', X) = \max_{\boldsymbol{w}' \in \mathcal{W}} \mathsf{cost}(\boldsymbol{w}', X) \leq (1+\epsilon)\mathsf{cost}(\hat{\boldsymbol{w}}, X) = (1+\epsilon)\mathsf{cost}(\mathcal{I}, X)$$

On the other hand,

$$\mathsf{cost}(\mathcal{I}', X) = \max_{\boldsymbol{w}' \in \mathcal{W}} \mathsf{cost}(\boldsymbol{w}', X) \geq \max_{\boldsymbol{w} \in \mathcal{W}} \mathsf{cost}(\boldsymbol{w}, X) - \epsilon\mathsf{cost}(\hat{\boldsymbol{w}}, X) = (1-\epsilon)\mathsf{cost}(\mathcal{I}, X).$$

2.4.2 EPAS for Socially Fair (k, z)-Clustering

In this section, we show how to use the coreset obtained from Theorem 2.4.1 to get a $(1+\epsilon)$-approximate solution to the Socially Fair (k, z)-Clustering problem and provide an EPAS with respect to k and d, when $|P|$ is small. By scaling the distances in the instance of Socially Fair (k, z)-Clustering, we assume that the distances are between 1 and Δ', for some number Δ'. Our algorithm (see Algorithm 2) uses the leader guessing idea of [40]. In the leader guessing approach, we guess the leader of every partition of a fixed optimal solution, where the leader of a partition is a closest point (client) in P to the corresponding optimal center. However, each point can participate in multiple groups, resulting in the total number of points being dependent on the number of groups, $|\mathcal{W}|$. But, we will show next that guessing the leaders from P without considering the groups in $\mathcal{W}$ is, in fact, sufficient. Further, to get a $(1+\epsilon)$-approximate solution, we use a standard ball decomposition lemma (e.g., use Lemma 2.4.4).

Theorem 2.4.3. *For any $0 < \epsilon \leq 1$, Algorithm 2, on input $\mathcal{I} = (F, P, \mathcal{W})$, computes $X \subseteq F : |X| \leq k$ such that $\mathsf{cost}(\mathcal{I}, X) \leq (1+\epsilon)\mathsf{OPT}(\mathcal{I})$ in time $\left((\frac{z}{\epsilon})^d \log n\right)^{O(k)} |P|^k \mathsf{poly}(n, m)$.*

Proof. First we bound the runtime of the algorithm. The leader enumeration (first forall loop) requires at most $|P|^k$ loops, one for each k-tuple of P. Assuming $\Delta' = \mathsf{poly}(n)$, the radii enumeration (second forall loop) for the leaders requires at most $\left(\frac{\log n/\epsilon}{\epsilon}\right)^{O(k)} = (\log n/\epsilon)^{O(k)}$ loops using discretized steps of size that is power of $(1 + \epsilon/10z)$. Finally, there are at most $(\frac{z}{\epsilon})^{O(dk)}$ many k-tuples of $T_1 \times \cdots \times T_k$ since $|T_i| = (\frac{z}{\epsilon})^{O(d)}$, yielding the claimed runtime.

For correctness, we will show that, for any group $\mathbf{w} \in \mathcal{W}$, we have $\mathsf{cost}(\mathbf{w}, X) \leq (1+\epsilon)\mathsf{cost}(\mathbf{w}, O)$, which implies $\mathsf{cost}(\mathcal{I}, X) \leq (1+\epsilon)\mathsf{cost}(\mathcal{I}, O)$, where $O = \{o_1, \cdots, o_k\}$ is an optimal solution center. Let Π_i be the set of points in P served by o_i, for $i \in [k]$. Let $\ell_i^* \in \Pi_i$ be a point that is closest to o_i. Let this distance be λ_i^*, i.e., $\lambda_i^* := \delta(\ell_i^*, o_k)$. We call ℓ_i^* as the leader of Π_i with radius λ_i^*. Let λ_i' be the smallest number equal

to some power of $(1+\epsilon/10z)$ that is larger than λ_i^*. Then, note that $\lambda'_i \geq \lambda_i^* \geq \frac{\lambda'_i}{(1+\epsilon/10z)}$. Next, consider the $\frac{\epsilon}{20z}$-ball decomposition of $\mathsf{ball}(\ell_i^*, \lambda'_i)$, and let b_i^* be a ball containing o_i and let t_i^* be its center. Now, consider the iteration of the algorithm corresponding to leader-radii enumeration $(\ell_1^*, \cdots, \ell_k^*)$ and $(\lambda'_1, \cdots, \lambda'_k)$ and, center enumeration $(t_1^*, \cdots, t_k^*)$. Then, for any $p \in \Pi_i, i \in [k]$, we have that

$$\delta(p, t_i^*) \leq \delta(p, o_i) + \delta(o_i, t_i^*) \leq \delta(p, o_i) + \frac{\epsilon}{10z}\lambda'_i \leq \delta(p, o_i) + \frac{\epsilon}{10z}\left(1 + \frac{\epsilon}{10z}\right)\lambda_i^* \leq \left(1 + \frac{\epsilon}{5z}\right)\delta(p, o_i).$$

Hence, for any group $\mathbf{w} \in \mathcal{W}$, we have $\mathbf{w}[p]\delta(p, t_i^*)^z \leq (1 + \frac{\epsilon}{5z})^z \mathbf{w}[p]\delta(p, o_i)^z \leq (1+\epsilon)\mathbf{w}[p]\delta(p, o_i)^z$, where we used the inequality $(1 + \frac{\epsilon}{5z})^z \leq e^{\epsilon/5} \leq 1 + \frac{\epsilon}{5} + (\frac{\epsilon}{5})^2 + \cdots \leq 1 + \epsilon$. Hence, $\mathsf{cost}(\mathbf{w}, X) = \sum_{p \in P} w[p]\delta(p, X)^z \leq (1+\epsilon)\mathsf{cost}(\mathbf{w}, O)$. □

We conclude this section by proving the main claim of this section (Theorem 1.3.3) by using the results of Theorem 2.4.1 and Theorem 2.4.3 as follows.

Proof of Theorem 1.3.3. Given an instance $\mathcal{I} = (F, P, \mathcal{W})$ of Socially Fair (k, z)-Clustering, and the accuracy parameter $\epsilon > 0$, we invoke Theorem 2.4.1 on $\mathcal{I}$ with parameter $\epsilon/10$ to obtain an coreset $(P', \mathcal{W}')$ such that $P' \subseteq P : |P'| = \left(\frac{2^z}{\epsilon}\right)^{O(d)} kz \log n$. Let $\mathcal{I}' = (F, P', \mathcal{W}')$ be the resulting instance. Then, we invoke Theorem 2.4.3 on $\mathcal{I}'$ with parameter $\epsilon/10$ to obtain $X \subseteq F : |X| \leq k$ such that $\mathsf{cost}(\mathcal{I}', X) \leq (1 + \epsilon/10)\mathsf{OPT}(\mathcal{I}')$.

First we analyze the overall running time. With $|P'| = \left(\frac{2^z}{\epsilon}\right)^{O(d)} kz \log n$, Theorem 2.4.3 runs in time $\left(\left(\frac{2^z}{\epsilon}\right)^d kz \log n\right)^{O(k)} \mathsf{poly}(n, m)$, leading to $\left(\left(\frac{2^z}{\epsilon}\right)^d zk \log k\right)^{O(k)} \mathsf{poly}(n, m)$ as the overall running time as desired. For correctness, consider

$$\begin{aligned}
\mathsf{cost}(\mathcal{I}, X) &\leq (1+\epsilon/10)\mathsf{cost}(\mathcal{I}', X) && \text{by the coreset property} \\
&\leq (1+\epsilon/10)^2\mathsf{OPT}(\mathcal{I}') && \text{by Algorithm 2} \\
&\leq (1+\epsilon/10)^3\mathsf{OPT}(\mathcal{I}) && \text{by the coreset property} \\
&\leq (1+\epsilon)\mathsf{OPT}(\mathcal{I}).
\end{aligned}$$

□

Algorithm 2: $(1+\epsilon)$-approximation algorithm for Socially Fair (k,z)-Clustering

Data: Instance $\mathcal{I} = (F, P, \mathcal{W})$ of Socially Fair (k,z)-Clustering
Result: $(1+\epsilon)$-approximate solution $X \subseteq F$

Let $X \leftarrow \emptyset$;
forall *k-tuples* $(\ell_1, \cdots, \ell_k)$ *of* P **do**
  forall *k-tuples* $(\lambda_1, \cdots, \lambda_k)$ *radii of* $(\ell_1, \cdots, \ell_k)$ *that are power of* $(1+\epsilon/10z)$ **do**
    for $i \in [k]$ **do**
      $\mathcal{B}_i \leftarrow \{\frac{\epsilon}{20z}$-ball decomposition of $\mathsf{ball}(\ell_i, \lambda_i)\}$;
    end
    $T_i \leftarrow \{f \in F \mid f$ is an arbitrary facility in ball $b \in \mathcal{B}_i\}$ [a] ;
    forall *k-tuples* $(t_1, \cdots, t_k)$ *of* $T_1 \times \cdots \times T_k$ **do**
      if $cost(\mathcal{I}, \{t_1, \cdots, t_k\}) < cost(\mathcal{I}, X)$ **then**
        $X \leftarrow \{t_1, \cdots, t_k\}$
      end
    end
  end
end
return X

[a] If $F = \mathbb{R}^d$ then $T_i \leftarrow \{x_b \in F \mid x_b$ is the center of ball $b \in \mathcal{B}_i\}$

Lemma 2.4.4 (Ball decomposition lemma)**.** *Consider a metric space (X, δ) with doubling dimension d. A subset $A \subseteq X$ is ϵ-dense in X, if $\forall x \in X$, $\exists y \in A$ such that $\delta(x,y) \leq \epsilon$. A is ϵ-separated, if $\forall x \neq y \in A$, $\delta(x,y) > \epsilon$ and A is ϵ-net of X if A is ϵ-separated as well as ϵ-dense. Then we have the following:*

1. *There exists an ϵ-dense set $A \subseteq \mathsf{ball}(x,r)$ of size $(\frac{r}{\epsilon})^{O(d)}$ $\forall x \in X$, $r > 0$, $\epsilon \leq r/2$.*
2. *For all ϵ-separated set $A \subseteq \mathsf{ball}(x,r)$, $r > 0$, $\epsilon \leq r/2$ it holds that $|A| \leq (\frac{r}{\epsilon})^{O(d)}$.*

Proof. 1. Let us denote k as the doubling constant of the considered metric space (X, δ) and $m = \lceil \log(r/\epsilon) \rceil$. Then by the definition, we have $k = 2^{O(d)}$. Note ball(x,r) can be covered with k balls of radius $r/2$. Further each of these balls of radius $r/2$ can be covered with k balls of radius $r/4$ resulting the original ball ball(x,r) can be covered with k^m balls of radius $\frac{r}{2^m}$ (note $\frac{r}{2^m} \leq \epsilon$). Again since $k = 2^{O(d)}$, a simple calculation shows that $k^m = (\frac{r}{\epsilon})^{O(d)}$. Let A be the centers of these balls then clearly A is ϵ-dense as required.

2. Note that the balls of radius $(\frac{\epsilon}{2})$ around the points of A are disjoint as the points in the set A have pairwise distance strictly greater than ϵ and further their union is included in the $\mathsf{ball}(x, r + \frac{\epsilon}{2})$. Hence, A is at most the size of any $\epsilon/2$-dense set within the $\mathsf{ball}(x, r + \frac{\epsilon}{2})$. By the previous claim, we finally prove that there exists an $\epsilon/2$-dense set in $\mathsf{ball}(x, r + \frac{\epsilon}{2})$ of size $(\frac{2r}{\epsilon})^{O(d)}$.
   One can even construct the set A greedily. Initially $A = \phi$. Next we choose an arbitrary point in ball(x,r) and add it to A. Let B denote the union of the closed balls of radius ϵ around the points in A. As long as we find a point $p \in \mathsf{ball}(x,r) \setminus B$, we add this particular point to the set A. Note, when the algorithm stops the resulting set A becomes an ϵ-net as both ϵ-dense as well as ϵ-separated conditions are being satisfied.

□

2.5 Conclusions and Open Problems

In this chapter we provide a detailed exploration of the parameterized approximability of Socially Fair (k, z)-Clustering problem in geometric spaces, with the number k of centers as the parameter. Our main results are as follows: (i) For a universal constant $\eta_0 > 0.0006$, we design a $3^z(1-\eta_0)$-factor FPT approximation algorithm for Socially Fair (k, z)-Clustering in *discrete* high-dimensional Euclidean spaces where the set of potential centers is finite. This shows that the lower bound of 3^z for general metrics by Goyal, Jaiswal [73] no longer holds when the metric has geometric structure. (ii) We show that Socially Fair (k, z)-Clustering in discrete Euclidean spaces is $(\sqrt{3/2}-o(1))$-hard to approximate for FPT algorithms, even if we consider the special case k-Center in logarithmic dimensions. This rules out a $(1+\epsilon)$-approximation algorithm running in time $f(k, \epsilon)\mathsf{poly}(m, n)$ (also called EPAS), giving a striking contrast with the EPAS for the *continuous* setting where centers can be placed anywhere in the space. (iii) we obtain an EPAS for Socially Fair (k, z)-Clustering in discrete Euclidean spaces when the dimension is sublogarithmic. Our EPAS works also for metrics of sub-logarithmic doubling dimension.

Our results also suggest directions for future research. One open question is whether we can design an approximation algorithm that is better than the current bound. Another direction is to investigate whether a stronger lower bound can be achieve for FPT approximation algorithms.

Chapter 3

EPAS for General Norm Clustering

In this chapter we present results for Norm k-Clustering. Even basic clustering problems such as k-Median, k-Center, and k-Means that have been researched for more than a half century and still they remain elusive from many perspectives of computation. This chapter is inspired by the following meta-question:

> For a given k-clustering objective and a (structured) metric space, does an EPAS exist?

3.1 Efficient Parameterized Approximation Schemes for Norm k-Clustering

As an input to the (general) k-clustering problem, we are given n data points P, candidate centers F, a metric space $M = (P \cup F, \delta)$, a positive integer k, and an objective function $f\colon \mathbb{R}^P \to \mathbb{R}$. When a set of k "open" centers $X \subseteq F$ is chosen, this solution induces a cost vector $\boldsymbol{\delta}(P, X) = (\delta(p, X))_{p \in P}$ where $\delta(p, X) = \min_{x \in X} \delta(p, x)$ represents the distance from point p to the closest center in X. Our goal is to minimize $f(\boldsymbol{\delta}(P, X))$. We call this problem the k-clustering problem with cost function f. We may think of the function f as "aggregating" the costs incurred by the points. For example, we can formulate basic k-clustering objectives via the functions $f(\boldsymbol{x}) = \sum_{p \in P} x(p)$ (k-Median), $f(\boldsymbol{x}) = \sum_{p \in P} x(p)^2$ (k-Means) and $f(\boldsymbol{x}) = \max_{p \in P} x(p)$ (k-Center).

Most natural and well-studied clustering objectives can be modeled using (a generalization of) the concept of *norm* optimization introduced by Chakrabarty and Swamy [29]. More specifically, we are interested in the setting where the objective f is a norm.

Definition 3.1.1 (Norm)**.** *A norm is a function $f\colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$, $n \in \mathbb{N}$ that satisfies (i) for all $\boldsymbol{x} \in \mathbb{R}^n$, $f(\boldsymbol{x}) = 0$ if and only if $\boldsymbol{x} = \boldsymbol{0}$, (ii) $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n\colon f(\boldsymbol{x} + \boldsymbol{y}) \leq f(\boldsymbol{x}) + f(\boldsymbol{y})$, and (iii) $\forall \boldsymbol{x} \in \mathbb{R}^n, \lambda \in \mathbb{R}\colon f(\lambda \boldsymbol{x}) = |\lambda| f(\boldsymbol{x})$.*

We say that f is *monotone* if $f(\boldsymbol{x}) \leq f(\boldsymbol{y})$ whenever $\boldsymbol{x} \leq \boldsymbol{y}$. By Norm k-Clustering we refer to the k-clustering problem whose objective $f\colon \mathbb{R}^P \to \mathbb{R}_{\geq 0}$ is a monotone norm. While Chakrabarty and Swamy [29] further require that f be symmetric[1], our algorithmic framework applies to all monotone norm cost functions.

[1] We say that f is symmetric if $f(\boldsymbol{x}) = f(\boldsymbol{x}')$ whenever $\boldsymbol{x}'$ can be obtained by reordering coordinates of $\boldsymbol{x}$.

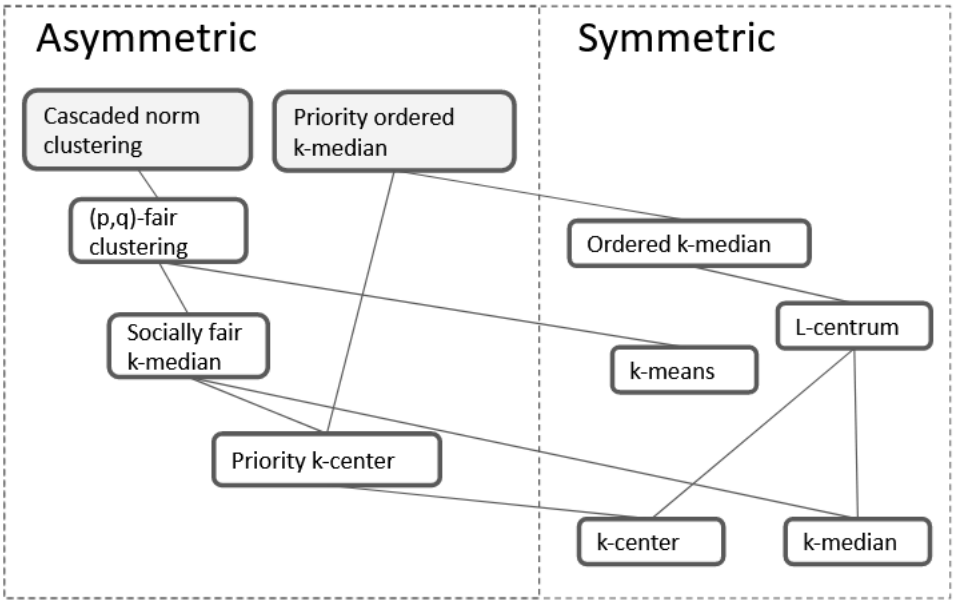


Figure 3.1: Selected clustering objectives that can be formulated as monotone norm minimization. The line illustrates generalization (bottom is a special case of top).

This family includes the following well-known clustering problems (see Figure 3.1 for an overview):

- **From k-Means, k-Center, and k-Median to (k,z)-Clustering**: All the basic clustering problems can be captured by the ℓ_z-norm when $z \in \{1, 2, \infty\}$. In fact, the (k,z)-clustering problem [86, 49, 45] (for constant positive integer z) uses the objective function $g(\boldsymbol{x}) = \sum_{p\in P} |x(p)|^z$. (This function itself is not a norm, but we can instead consider the ℓ_z-norm $f(\boldsymbol{x}) = g(\boldsymbol{x})^{1/z}$.)

- **Weighted k-Center (or Priority k-Center)**: The weighted version of k-Center [100, 14, 109] generalizes the k-Center so that each data point $p \in P$ is associated with a positive weight (or priority) $w(p)$, and the objective is to minimize the (weighted) maximum distance to a center.[2] This problem can be modelled by the "weighted max" norm $f(\boldsymbol{x}) = \max_{p\in P} w(p)x(p)$. One can analogously define the weighted versions of k-Median and k-Means (see, for example, [41]). We remark that the underlying weighted norms are not symmetric.

- **ℓ-Centrum**: This problem (sometimes called k-Facility ℓ-Centrum) aims to minimize the sum of the connection costs among the ℓ "most expensive" points (that is, those that are furthest away from the open centers). The problem generalizes both k-Center ($\ell = 1$) and k-Median ($\ell = |P|$) problem [117]. (See the books [107, 96] for more details on ℓ-Centrum and the more general Ordered k-Median discussed

[2]For convenience of presentation, the terminologies we use are somewhat different from the literature.

below.) This problem can be modelled by the top-ℓ norm $f(\boldsymbol{x}) = \sum_{j=1}^{\ell} x^{\downarrow}(j)$ where $\boldsymbol{x}^{\downarrow}$ denotes the reordering of vector $\boldsymbol{x}$ so that the entries appear non-increasingly. The top-ℓ norm is symmetric.

- **Ordered k-Median**: This problem further generalizes ℓ-Centrum, allowing flexible penalties to be applied to data points that incur the highest connection costs. More formally, the objective is the *ordered weighted norm* $f(\boldsymbol{x}) = \boldsymbol{v}^{\intercal}\boldsymbol{x}^{\downarrow}$ where $\boldsymbol{v} \in \mathbb{R}^n_{\geq 0}$ is a non-increasing cost vector, that is, $v(1) \geq v(2) \geq \ldots \geq v(n)$. ℓ-Centrum corresponds to $\boldsymbol{v} = (1, \ldots, 1, 0, \ldots 0)$ where the first ℓ-entries of $\boldsymbol{v}$ are ones. This problem has already received attention for a few decades [27, 29, 24]. We remark that the f here is a monotone and symmetric norm.

- **Socially Fair k-Median (or Robust k-Median)**: In Socially Fair k-Median, along with the point set P, we are given m different (not necessarily disjoint) subgroups such that $P = \bigcup_{i\in[m]} P_i$. Our goal is to find a set X of centers that incurs fair costs to the groups by minimizing the maximum cost over all the groups. In other words,

$$\min_{\substack{X\subseteq F\\ |X|=k}} \max_{i\in[m]} \sum_{p\in P_i} \delta(p, X)\,.$$

  Due to distinct applications in at least two domains, this variant of clustering has recently been studied extensively: (i) in algorithmic fairness [5, 73, 103, 70] and (ii) in the robust optimization context, this problem is known as Robust k-Median, which intends to capture the applications when we are uncertain about the actual data scenarios (corresponding to the groups P_i) that may come up [10, 21, 19]. The resulting norm is generally asymmetric.

- **(z, q)-Fair Clustering:** Our problem also models a clustering problem called (z, q)-Fair Clustering[3] introduced by Chlamtáč et al. [31], which generalizes Socially Fair k-Median.

  In particular, one can view the cost function f of Socially Fair k-Median as a "two-level" aggregate cost: First, cost $\sum_{p\in P_i} \delta(p, X)$ incurred by group P_i, $i \in [m]$ can be viewed as weighted ℓ_1-norm $\boldsymbol{w}_i^{\intercal}\boldsymbol{x}$ where $\boldsymbol{w}_i = \mathbf{1}_{P_i} \in \{0, 1\}^P$ denotes the characteristic vector of P_i. Second, these group costs are further aggregated through ℓ_∞, that is, $f(\boldsymbol{x}) = \max(\boldsymbol{w}_1^{\intercal}\boldsymbol{x}, \boldsymbol{w}_2^{\intercal}\boldsymbol{x}, \ldots, \boldsymbol{w}_m^{\intercal}\boldsymbol{x})$.

  (z, q)-Fair Clustering allows arbitrary uses of ℓ_z and ℓ_q norms to aggregate the costs in two levels. The cost function is defined as $f(\boldsymbol{x}) = g(\boldsymbol{h}(\boldsymbol{x}))$ where g is any ℓ_q-norm function and $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_m(\boldsymbol{x}))$ where $h_i(\boldsymbol{x})$ is a weighted ℓ_z-norm, that is, $h_i(\boldsymbol{x}) = \left(\sum_{p\in P} w_i(p)x(p)^z\right)^{1/z}$ for arbitrary weight vectors $\boldsymbol{w}_i \in \mathbb{R}^P_{\geq 0}$, $i \in [m]$. It is easy to check that $f(\boldsymbol{x}) = g(\boldsymbol{h}(\boldsymbol{x}))$ is a monotone norm whenever g and $\{h_i\}$ are. The objective is generally an asymmetric norm.

- **Beyond the Known Problems**: Our (asymmetric) norm formulation allows us to model more complex clustering objectives that might be useful in some application settings and, to our knowledge, have not

[3] Chlamtáč et al. [31] call the problem (p, q)-Fair Clustering. For the sake of consistency with the notation in the rest of the paper, we changed the naming slightly.

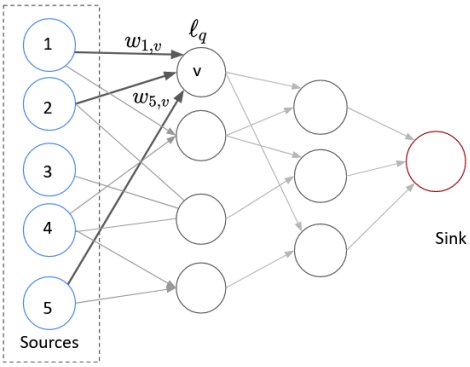


Figure 3.2: The DAG here describes evaluation of function f. Node v is labeled with the ℓ_q norm, so the evaluation at node v is $\eta(v) = (w_{1,v}x_1^q + w_{2,v}x_2^q + w_{5,v}x_5^q)^{1/q}$.

yet been considered in the algorithms community. One such objective is Priority Ordered k-Median: We have the cost function $f(\boldsymbol{x}) = \boldsymbol{v}^\intercal \boldsymbol{x}_{\boldsymbol{w}}^\downarrow$ where the weight vector $\boldsymbol{v} \in \mathbb{R}^n_{\geq 0}$, and priority vector $\boldsymbol{w} \in \mathbb{R}^P_{\geq 0}$ are given as input, and where $\boldsymbol{x}_{\boldsymbol{w}} = (w(p)x(p))_{p\in P}$. This objective generalizes both Priority k-Center and Ordered k-Median. Another natural objective is the (multi-level) Cascaded Norm Clustering, which generalizes (z,q)-Fair Clustering to allow multiple levels of cost aggregation. The cost function f for this problem is described by a directed acyclic graph (DAG) D with one sink node and $|P|$ source nodes (each source corresponds to a point in P). Each non-source node v is associated with a norm ℓ_q for some q, and each edge (u,v) has weight $w_{u,v}$. Given such a DAG D, the value of $f(\boldsymbol{x})$ can be evaluated by computing the evaluations at nodes in $V(D)$ in (topological) order from sources to sink: (i) The evaluation at source $p \in P$ is $\eta(p) = x(p)$, (ii) For any non-source node $v \in V(D)$ labelled with the norm ℓ_q, we evaluate $\eta(v) = \left(\sum_{u\in N^-(v)} w_{u,v}\eta(u)^q\right)^{1/q}$, and (iii) the value of $f(\boldsymbol{x})$ is the evaluation of the sink. See Figure 3.2 for illustration. (z,q)-Fair Clustering is a special case when D has 3 layers with the middle layer using the same norm. Of course, also other basic monotone norms such as top-ℓ or ordered weighted norms could be composed to more complex norms analogously.

3.2 Overview of Techniques

In this section, we give an informal overview of the technical ideas appearing in the chapter. The main result will be built step by step: we believe that it is already interesting to understand our main result specialized to Weighted k-Center and Weighted k-Median. Our starting point is the EPAS of Badǒiu et al. [13] for unweighted k-Center that works on high-dimensional Euclidean spaces. We redesign and change this algorithm in order to be able to present it with a clean division into two parts: a simple branching algorithm and a bound on the abstract concept of (algorithmic) ϵ-scatter dimension. This way, we obtain a sharp separation between the branching algorithm, which is specific to the objective and the bound on ϵ-scatter

dimension, which is specific to the metric. This can be contrasted with techniques based on coresets, which are inherently specific both to a single objective and to a single metric. The main message of the paper is that, with the right combination of additional ideas, this framework can be significantly generalized both in terms of objectives and metric spaces.

This section presents the main algorithmic ideas in three steps.

1. The algorithm for unweighted k-Center can be generalized to Weighted k-Center in a not completely obvious way.
2. Building on the algorithm for Weighted k-Center, we can solve Weighted k-Median with a preprocessing and a random selection step.
3. The Weighted k-Median algorithm can be generalized to arbitrary monotone norms by considering infinitely many Weighted k-Median instances defined by the subgradients.

While some of the challenges on the way may appear to have other approaches promising at first glance, we want to emphasize that it is nontrivial to find the combination of ideas that can be integrated together to obtain our main result. In particular, for Weighted k-Median the initial upper bounds have to be defined carefully in a way that allows, at the same time, an efficient random selection step and generalization to arbitrary monotone norms.

Weighted k-Center with Bounded Number of Different Weights Our starting point is a simple branching algorithm that is inspired by the EPAS of Badǒiu et al. [13] for unweighted k-Center. Instead of branching, it will be more convenient for us to present it as a randomized algorithm. Furthermore, we consider the more general setting of Weighted k-Center: the objective is to find a set O of k centers that minimizes $\max_p w(p)\delta(p, O)$. Let us first present the algorithm with the simplifying assumption that w is a weight function on the points whose range contains only at most τ different values. The unweighted problem corresponds to $w(p) = 1$ for every $p \in P$ and hence $\tau = 1$. It will be convenient to assume that we (approximately) know the value of OPT.

We start with k arbitrarily chosen candidates $X = \{x_1, \ldots, x_k\}$ for the k centers. We additionally introduce k sets of *requests* $Q_1, \ldots, Q_k$, where each request is of the form (p, r) with a point $p \in P$ and radius $r > 0$. For every $\kappa \in [k]$, we impose the *cluster constraint* requiring that, for every $(p, r) \in Q_\kappa$, center x_κ should be at distance at most r from p. Initially, we set $Q_\kappa = \emptyset$ for every κ, which means that these conditions are trivially satisfied. If we have $\max_p w(p)\delta(p, X) \leq (1 + \epsilon)\mathsf{OPT}$, then we can stop, as we have a $(1 + \epsilon)$-approximate solution at our hands. Otherwise, we have a point p with $\delta(p, X) > (1 + \epsilon)\mathsf{OPT}/w(p)$, while it is at distance at most $\mathsf{OPT}/w(p)$ from some center of a hypothetical optimum solution O. Thus the algorithm selects a $\kappa \in [k]$ uniformly at random, hoping it to be the index of the center that is at distance at most $\mathsf{OPT}/w(p)$ from p in the optimum solution O. Then we introduce the request $(p, \mathsf{OPT}/w(p))$ into the set Q_κ and select x_κ to be a center that satisfies the cluster constraint defined by all the requests in the updated Q_κ. Observe that if every random choice was compatible with the hypothetical optimum solution O, then the algorithm is

always able to find such a center, as the requests in Q_κ are always satisfied by the κ-th center of the optimum solution O.

We claim that if the ϵ-scatter dimension of the metric is bounded, then this algorithm stops after a bounded number of steps, either by finding an approximate solution or by failing to find a center satisfying the cluster constraints of some Q_κ. Let $x_\kappa^{(1)}, \ldots, x_\kappa^{(\ell)}$ be the different candidates for the κ-th center throughout this branch. Let $(p_\kappa^{(1)}, r_\kappa^{(1)}), \ldots, (p_\kappa^{(\ell)}, r_\kappa^{(\ell)})$ be the requests introduced to Q_κ: that is, for $1 \leq j \leq \ell$, the center $x_\kappa^{(j)}$ was chosen to be at distance at most $r_\kappa^{(i)}$ from every $p_\kappa^{(i)}$ for $1 \leq i < j$, but later was found to be at distance at least $(1+\epsilon)r_\kappa^{(j)}$ from $p_\kappa^{(j)}$. As there are at most τ different weights in the input, at least $\ell' = \ell/\tau$ of these requests have the same radius. That is, there is a subsequence $(x_\kappa^{(s_1)}, p_\kappa^{(s_1)}, r_\kappa^{(s_1)}), \ldots, (x_\kappa^{(s_{\ell'})}, p_\kappa^{(s_{\ell'})}, r_\kappa^{(s_{\ell'})})$ where every $r_\kappa^{(s_j)}$ for $j \in [\ell']$ is the same value $r \geq 0$. This means that we have a subsequence $(\bar{x}_1, \bar{p}_1), \ldots, (\bar{x}_{\ell'}, \bar{p}_{\ell'})$ with the property that $\delta(\bar{x}_i, \bar{p}_i) > (1+\epsilon)r$, but $\delta(\bar{x}_i, \bar{p}_j) \leq r$ for every $i < j$. By scaling down every distance by a factor of r, this is precisely an ϵ-scattering of length ℓ'. If we consider a class of metrics closed under scaling where the ϵ-scatter dimension is $\lambda(\epsilon)$, then this sequence cannot have length longer than $\lambda(\epsilon)$, implying that $\ell \leq \tau \cdot \lambda(\epsilon)$. We can conclude that the algorithm can introduce at most $\tau \cdot \lambda(\epsilon)$ requests into each Q_κ, hence the algorithm cannot perform more than $k \cdot \tau \cdot \lambda(\epsilon)$ iterations.

If every step of the algorithm randomly chooses an index $\kappa \in [k]$ that is consistent with the optimum solution O, then the only way it can stop is by finding an approximate solution. Therefore, the algorithm is successful with probability at least $q = k^{-k\cdot\tau\cdot\lambda(\epsilon)}$. The success probability can be boosted to be a constant arbitrarily close to 1 by the standard technique of repeating the algorithm $O(1/q)$ times, leading to a running time of $k^{k\cdot\tau\cdot\lambda(\epsilon)} \cdot \mathsf{poly}(n)$.

Weighted k-Center with Arbitrary Weights We show now how the algorithm can be extended to work in the weighted setting with arbitrary weights. Let us observe first that if there is no bound on the number τ of different weights, then we cannot bound the number of requests to a given Q_κ, even in very simple metric spaces such as $\mathbb{R}^1$. Suppose for example that the requests arriving to Q_κ are $(p^{(i)}, (1+2\epsilon)^{-i})$ for $i = 1, 2, \ldots$, where every $p^{(i)}$ is at the origin (or maybe within a very small radius of the origin). Then a center $x^{(i)}$ at $(1+2\epsilon)^{1-i}$ satisfies the first $i-1$ requests, but violates the constraint of the i-th by more than a $(1+\epsilon)$-factor. This sequence can be arbitrarily long, and the existence of such a sequence shows that we cannot bound the number of requests arriving to Q_κ if we don't have a bound on the number of different weights. Nevertheless, we show that the number of requests can be bounded if we start the algorithm by carefully seeding the initial requests. Let us remark that we know other simple modifications that achieve such a bound, but the technique described below turns out to be the one that can be extended further for Weighted k-Median and general norms.

The main idea is to bootstrap our algorithm with a constant-factor approximation. A simple greedy 3-approximation can be obtained following the ideas of Plesník [109]. Let us consider all the balls $\mathsf{ball}(p, \mathsf{OPT}/w(p))$ for every $p \in P$. Let us consider these balls in a nondecreasing order of radius, and mark each ball that does not intersect any of the balls marked earlier; let $\mathsf{ball}(p_\kappa, \mathsf{OPT}/w(p_\kappa))$, $1 \leq \kappa \leq k'$ be the marked balls. We should have $k' \leq k$: otherwise, we have more than k pairwise disjoint balls and each of

them has to contain a center of the solution, contradicting the assumption that value OPT can be achieved with k centers. For $1 \leq \kappa \leq k'$, let x_i be any center in $\mathsf{ball}(p_\kappa, \mathsf{OPT}/w(p_\kappa))$ and let $Q_\kappa = \{(p_\kappa, \mathsf{OPT}/w(p_\kappa)\}$. For $k' < \kappa \leq k$, we choose x_i arbitrarily and let $Q_\kappa = \emptyset$. Let us observe that with this definition of the Q_κ's, we have $\delta(p, X) \leq 3\mathsf{OPT}/w(p)$ during *every iteration* of our algorithm. Indeed, if the ball of p was marked, then X always contains a center in $\mathsf{ball}(p_\kappa, \mathsf{OPT}/w(p_\kappa))$; if the ball of p was unmarked, then it intersects a marked ball with not larger radius that contains a center of X.

The main claim is that the ratio between the radii of two requests appearing in Q_κ can be bounded by $O(1/\epsilon)$. Suppose that (p, r) and (p', r') are two requests in Q_κ (introduced in any order) and we have $r' < \epsilon r/4$. A center of the optimum solution satisfies both request, hence we have $\delta(p, p') \leq r + r'$. As shown above, at every step of the algorithm there is a center in X at distance at most $3r'$ from p'; let y be such a center at the step when request (p, r) was introduced. Then we have

$$\delta(p, y) \leq \delta(p, p') + \delta(p', y) \leq r + r' + 3r' \leq (1 + \epsilon)r,$$

contradicting the need for the first request.

We can use the standard assumption that every weight is of the form $(1 + \epsilon)^i$ for some integer i: by rounding down every weight to the largest number of this form, we change the objective only by a factor of $1 + \epsilon$. If every weight is of the form $(1 + \epsilon)^i$, then the $O(1/\epsilon)$ bound proved above implies that the requests introduced into Q_κ for some fixed $\kappa \in [k]$ have $O(1/\epsilon \cdot \log 1/\epsilon)$ different radii. Therefore, we can bound the total number of requests (and hence the number of iterations) by $O(\lambda(\epsilon) \cdot k/\epsilon \cdot \log 1/\epsilon)$. This leads to a $k^{O(\lambda(\epsilon)\cdot k/\epsilon \cdot \log 1/\epsilon)} \cdot \mathsf{poly}(n)$ time randomized algorithm with constant success probability.

From Weighted k-Center to Weighted k-Median Towards our goal of understanding general norms, let us consider now the Weighted k-Median problem, where the objective is to find a set O of k centers that minimize $\sum_p w(p)\delta(p, O)$. We will try to solve this problem by interpreting it as a Weighted k-Center problem on a weighted point set that we dynamically discover during the course of the algorithm.

We would like to turn the *linear constraint* $\sum_p w(p)\delta(p, X) \leq \mathsf{OPT}$ of Weighted k-Median into a *distance constraint:* some point p should be at distance at most r to the solution. Let X be the current solution and suppose that $\sum_p w(p)\delta(p, X) > (1 + \epsilon)\mathsf{OPT}$. The intuition is that $\sum_p w(p)\delta(p, X) > (1 + \epsilon) \sum_p w(p)\delta(p, O)$ for an optimum solution O implies that a nontrivial fraction of the points should satisfy $\delta(p, X) > (1 + \epsilon/3)\delta(p, O)$, that is, their distances to the solution has to be improved by more than a factor of $1 + \epsilon/3$. More precisely, an easy averaging argument shows if we select a point p with probability proportional to $w(p)\delta(p, X)$, then p satisfies $\delta(p, X) > (1 + \epsilon/3)\delta(p, O)$ with probability $\Omega(\epsilon)$. We call such a point p an $\epsilon/3$-*witness,* certifying that the current solution has to be improved.

Assuming that the sampled point p is indeed a $\epsilon/3$-witness, we proceed as in the case of Weighted k-Center. We randomly choose an index κ and introduce the request $(p, \delta(p, X)/(1 + \epsilon/3))$ into Q_κ, to update the cluster constraint by requiring that x_κ should be closer to p than in the current solution. If there is a center satisfying all the requests in Q_κ, then we update x_κ. These steps are repeated until we arrive to a

solution P with $\sum_p w(p)\delta(p,X) \leq (1+\epsilon)\mathsf{OPT}$.

In each step, with probability $\Omega(\epsilon/k)$, the algorithm chooses an $\epsilon/3$-witness p and a center κ that is consistent with some hypothetical optimum solution O. However, it is not clear how to bound the running time of the algorithm. It can happen that the requests arriving to Q_κ have smaller and smaller radii. As we have seen for Weighted k-Center, in such a scenario we cannot bound the number of steps even in $\mathbb{R}^1$ It is crucial to have some control on the sequence of radii that appear in the requests. Therefore, next we show how to ensure that the radii in the requests to center κ stay within a bounded range.

Initial Upper Bounds For each point p, we compute a weak upper bound $u(p) \geq \delta(p,O)$ on the distance to the optimum solution. Then instead of starting with an arbitrary set of k centers, we bootstrap the algorithm by a solution approximately satisfying all these upper bounds. We argue that this can be done in such a way that ensures that the radii appearing in the requests to each center κ stay within a bounded range.

If a point p^* has weight $w(p^*)$, then $u(p^*) = \mathsf{OPT}/w(p^*)$ is an obvious upper bound on the distance of p^* to O: otherwise, we would have $\sum_p w(p)\delta(p,O) \geq w(p^*)\delta(p^*,O) > \mathsf{OPT}$. This bound was sufficient for the Weighted k-Center problem, but the nature of Weighted k-Median allows us to get much stronger upper bounds in many cases. For example, if there are c points of the same weight w roughly at the same position, then each of them should be at distance at most $\mathsf{OPT}/(wc)$ from O. Indeed, otherwise the total contribution of these c points to the sum would be greater than OPT. More generally, if there is a radius r such that total weight of the points at distance at most r from p is at least OPT/r, then we claim that p is at most distance $2r$ from O. Indeed, otherwise all these points would be at distance more than r from O, making their total contribution greater than OPT. Therefore, we can define $u(p) = 2r$, where r is the smallest radius with the property that the total weight of the points at distance at most r from p is at least OPT/r. Note that $u(p)$ can be determined in polynomial time from the weights of the points and their distance matrix.

Similarly to our Weighted k-Center algorithm, we start with a 3-approximation of the constraints given by the upper bounds $u(p)$ for $p \in P$. Let us go through the points in a nondecreasing order of $u(p)$ and let us greedily choose a maximal independent set of the balls $\mathsf{ball}(p,u(p))$. We should find at most k such balls. Let us choose a center in each ball; it is easy to see that every point p has a selected center at distance at most $3u(p)$ from it. If center x_κ was selected to be a center in $\mathsf{ball}(p,u(p))$, then we initialize Q_κ with the request $(p,u(p))$. This ensures that during every step of the algorithm, it remains true that every point p is at distance at most $3u(p)$ from the current solution.

We run the algorithm for Weighted k-Median with this initial solution. Before analyzing the algorithm, let us make a nontrivial change in the random selection. We have seen that with probability $\Omega(\epsilon/k)$, we select a random point p and $\kappa \in [k]$ such that $\delta(p,X) \geq (1+\epsilon/3)\delta(p,O)$ for some optimal solution O. A key claim of the proof is that with probability $\Omega(\epsilon/k)$, it is also true that $u(p) \leq 2k\delta(p,X)/\epsilon$ (see Lemma 3.5.10). Intuitively, the total contribution of the $\epsilon/3$-witnesses that are too close to some center $x_\kappa \in X$ cannot be very large, because then all of these witnesses would be in a small ball, implying that the upper bound $u(p)$ should be smaller. Note that this is the point in the proof where we crucially utilize the exact definition of $u(p)$. With this claim at hand, we can modify the algorithm such that we are randomly choosing a point p satisfying

$u(p) \leq 2k\delta(p, X)/\epsilon$, with probability proportional to $w(p)\delta(p, X)$. It remains true that p is an $\epsilon/3$-witness with probability $\Omega(\epsilon/k)$.

Let us analyze now the algorithm and bound the number of times a center x_κ is updated. We want to argue that the radius in the requests remains in a bounded range. Suppose that we update cluster κ with requests (p, r) and (p', r') (in either order) such that $r' \ll \epsilon^2 r/k$. If the algorithm does not fail, then there is a center x_κ satisfying both requests. By the triangle inequality, this means that the $\delta(p, p') \leq r + r' < r + \epsilon r/6$. Furthermore, by the constraint $u(p') \leq 2k\delta(p', X)/\epsilon = 2k(1+\epsilon/3)r'/\epsilon$ on our selection of the random point p', we have that $u(p')$ is much smaller than $\epsilon r/18$. At every step of the algorithm, the upper bound $u(p')$ is 3-approximately satisfied by the current solution X. Thus there should be a center in X much closer than $\epsilon r/6$ to p'. Together with $\delta(p, p') < r + \epsilon r/6$, it follows that there is always a center in X at distance at most $(1 + \epsilon/3)r$ from p, contradicting the need for the request (p, r).

Thus the combination of the two facts that (1) the upper bounds are always satisfied approximately and that (2) the radius in the request is not much smaller than the upper bound implies that the radius in the requests stays within a bounded range. Then we can argue as in the case of the Weighted k-Center problem. If every weight is rounded to a power of $(1 + \epsilon)$, then each cluster is given requests with only a bounded number of different radii. If many requests arrive, then there is a long subsequence of the requests with the same radius. This means that the bound on the ϵ-scatter dimension can be used to bound the length of this subsequence, and hence the total number of requests to all clusters.

From Weighted k-Median to General Norms Using Subgradients Next we show how to solve the clustering problem for an arbitrary monotone norm by interpreting it as collection of Weighted k-Median instances that we need to satisfy simultaneously. We will repeatedly solve such Weighted k-Median instances that are dynamically discovered during the course of the algorithm.

It will be convenient to use the notion of subgradients. For our purposes, it is sufficient to discuss subgradients in the context of a monotone norm $f : \mathbb{R}^n \to \mathbb{R}$. We say that $\boldsymbol{g}$ is a *subgradient* of f at point $\boldsymbol{x}$ if $f(\boldsymbol{x}) = \boldsymbol{g}^\intercal \boldsymbol{x}$ and $f(\boldsymbol{y}) \geq \boldsymbol{g}^\intercal \boldsymbol{y}$ for every $\boldsymbol{y} \in \mathbb{R}^n$. It is known that every monotone norm has a nonnegative subgradient $\boldsymbol{g} \geq 0$ at every point $\boldsymbol{x} \geq 0$. Checking whether a vector $\boldsymbol{g}$ is a subgradient at $\boldsymbol{x}$ and finding a subgradient at $\boldsymbol{x}$ can be formulated as convex optimization problems, hence can be (approximately) solved using the ellipsoid method if f can be efficiently computed [78].

Suppose that we have a current solution X and let $\boldsymbol{x} \in \mathbb{R}^P_{\geq 0}$ be the vector representing the distances of the points in P to X. Suppose that X is not (approximately) optimal: $f(\boldsymbol{x}) > (1 + \epsilon)\mathsf{OPT}$. Let us compute a sugradient $\boldsymbol{g}$ of f at $\boldsymbol{x}$; we have $\boldsymbol{g}^\intercal \boldsymbol{x} = f(x) > (1 + \epsilon)\mathsf{OPT}$ and $\boldsymbol{g}^\intercal \boldsymbol{y} \leq f(\boldsymbol{y}) = \mathsf{OPT}$ for the optimum solution $\boldsymbol{y}$. That is, $\boldsymbol{g}^\intercal \boldsymbol{x} \leq \mathsf{OPT}$ is a linear constraint satisfied by the optimum solution and violated by the current solution. Then defining the weights $w(p)$ based on the coordinates of $\boldsymbol{g}$ gives an instance of Weighted k-Median, with $\sum_p w(p)\delta(p, X) > (1 + \epsilon)\mathsf{OPT}$ for the current solution X. Now we can proceed as above for the Weighted k-Median problem: we randomly choose a point p and cluster κ, introduce a new request into Q_κ, find a new center x_κ, etc., until we arrive to a solution X with $\sum_p w(p)\delta(p, X) \leq (1 + \epsilon)\mathsf{OPT}$. If this new solution X is still nonoptimal for the original norm problem, that is, $f(\boldsymbol{x}) > (1 + \epsilon)\mathsf{OPT}$, then we

can again compute a subgradient, find a violated linear constraint (possibly the same as in the previous step). We repeat this until we find a solution with $f(\boldsymbol{x}) \leq (1+\epsilon)\mathsf{OPT}$.

Defining the upper bounds and bootstrapping the algorithm with a solution approximately satisfying the upper bounds were crucial for the analysis of the Weighted k-Median algorithm. For general norms, we can again define the upper bounds once we have the weights w based on the violated linear constraint $\boldsymbol{g}^\intercal \boldsymbol{x} \leq \mathsf{OPT}$. However, these upper bounds would not be useful for the analysis, as they would depend on the violated linear constraint, hence would change during the algorithm.

Intuitively, we can see the constraint $f(\boldsymbol{x}) \leq \mathsf{OPT}$ as an infinite number of Weighted k-Median instances, corresponding to the linear constraints $\boldsymbol{g}^\intercal \boldsymbol{x} \leq \mathsf{OPT}$ for *every* subgradient $\boldsymbol{g}$ of f. We would like to define $u(p)$ to be the smallest possible upper bound that can be assigned to p among all of these infinitely many Weighted k-Median instances. Determining this value seems to be a difficult task, but actually the answer is very simple. Recall that $u(p)$ was defined as twice the smallest r such that $\mathsf{ball}(p,r)$ contains total weight at least OPT/r. Thus to define the upper bound $u(p)$, we need to know what the maximum weight of the points in $\mathsf{ball}(p,r)$ can be among the infinitely many instances corresponding to all the subgradients. Let $\boldsymbol{b}$ be the characteristic vector of $\mathsf{ball}(p,r)$ (i.e., every coordinate is 1 or 0, depending on whether a point is in or not in the ball). Then the question is to determine the maximum of $\boldsymbol{g}^\intercal \boldsymbol{b}$ among all subgradients $\boldsymbol{g}$. It is easy to see that this maximum is exactly $f(\boldsymbol{b})$: if $\boldsymbol{g}$ is a subgradient at $\boldsymbol{b}$, then $\boldsymbol{g}^\intercal \boldsymbol{b} = f(\boldsymbol{b})$; if $\boldsymbol{g}$ is a subgradient at an arbitrary point $\boldsymbol{y}$, then $\boldsymbol{g}^\intercal \boldsymbol{b} \leq f(\boldsymbol{b})$. Thus we can determine the maximum weight of any ball and define the upper bounds accordingly. With these definitions, the analysis of the Weighted k-Median algorithm go through for general mononote norms. The two main properties of the upper bounds remain valid: (1) the upper bounds are satisfied by the optimum solution and (2) we can restrict our random choice of p to points where the distance to the solution is not much smaller than $u(p)$.

In summary, the final algorithm consists of the following steps (see Figure 3.3). First we compute the upper bounds $u(p)$ and greedily find a 3-approximate solution satisfying these constraints. Then we repeat the following steps until we reach a solution X for which the distance vector $\boldsymbol{x}$ satisfies $f(\boldsymbol{x}) \leq (1+\epsilon)\mathsf{OPT}$. We compute a subgradient $\boldsymbol{g}$ of f at $\boldsymbol{x}$ to obtain a violated linear constraint $\boldsymbol{g}^\intercal \boldsymbol{x} \leq \mathsf{OPT}$. We randomly choose a point p (according to the distribution described above) and require that p be at most distance $\delta(p,X)/(1+\epsilon/3)$ from the solution, that is, we obtain a violated distance constraint. Then we randomly choose a cluster $\kappa \in [k]$ and require that this distance constraint be satisfied by center x_κ. Thus we put the request $(p, \delta(p,X)/(1+\epsilon/3)$ into Q_κ find a new x_κ that satisfy the cluster constraints imposed by the requests in Q_κ, if possible. We repeat these steps until we arrive to a solution X with distance vector $\boldsymbol{x}$ satisfying $f(\boldsymbol{x}) \leq (1+\epsilon)\mathsf{OPT}$. Our analysis shows that each step is consistent with a hypothetical optimum solution O with probability $\Omega(\epsilon/k)$. Moreover, if ϵ-scatter dimension is bounded, then the algorithm has to find a solution or fail after a bounded number of iterations.

(Algorithmic) ϵ-Scatter Dimension After the general algorithm capable of handling any monotone norm objective, our second main contribution is bounding the ϵ-scatter dimension of various classes of metrics (Section 3.6). In the interest of space, we do not go into the details of these (mostly combinatorial) proofs,

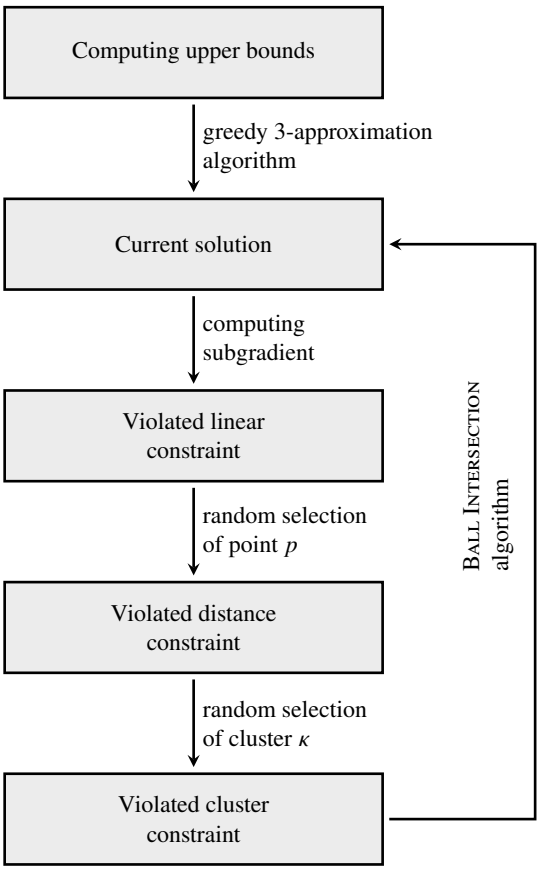


Figure 3.3: Overall structure of the main algorithm.

but give only a brief overview.

- **Bounded Doubling Dimension.** As outlined in the introduction, the set of points as well as the set of centers in an ϵ-scattering both form an ϵ-packing of a unit ball implying that any metric of doubling dimension d has ϵ-scatter dimension $(1/\epsilon)^{O(d)}$. See Theorem 1.3.6.

- **Bounded-Treewidth Graph Metrics.** The ϵ-scatter dimension bound for metrics defined by the shortest path metric of bounded-treewidth graphs is obtained by a delicate combinatorial proof that exploits both structure of the graph and properties of the ϵ-scattering. The bound we obtain is $\mathtt{tw}^{1/\epsilon^{O(\mathtt{tw})}}$ for graphs of treewidth $\mathtt{tw}$, that is, double exponential in $\mathtt{tw}$ for fixed ϵ. It remains is an interesting open question if this bound can be improved.

- **Planar Graph Metrics.** As outlined in the introduction, we can employ a known metric embedding result to reduce the problem of bounding the ϵ-scatter dimension of planar graphs to bounding the ϵ-scatter dimension of bounded-treewidth graphs. In particular, the result by Fox-Epstein, Klein, and

Schild [64] provides an (approximate) metric embedding of planar metrics into low-treewidth metrics, which can be used to obtain a $2^{2^{\mathsf{poly}(1/\epsilon)}}$ bound on the ϵ-scatter dimension of planar graph metrics.

- **Continuous High-Dimensional Euclidean Space.** As mentioned in the introduction, the high-dimensional Euclidean space does not have bounded ϵ-scatter dimension. However, in the continuous Euclidean space, where any point of the space can be a center, we can bound the *algorithmic ϵ-scatter dimension*. Towards this, we replace the center player by an algorithmic "player" applying the algorithm by Kumar and Yildirim [95] for Weighted 1-Center. To achieve bounded algorithmic ϵ-scatter dimension, this algorithm would require, however, a bounded aspect ratio of the radii in the input requests. We therefore prove an aspect-ratio condition (which holds even for general metrics) implying that it is sufficient for the algorithm to handle instances with aspect-ratio $O(1/\epsilon)$. We combine this result with the algorithm by Kumar and Yildirim to prove bounded algorithmic ϵ-scatter dimension for continuous high-dimensional Euclidean space, that is, Theorem 1.3.9.

3.3 Preliminaries

Subgradients of Norms We state definitions and summarize basic facts about subgradients of norms that we will use throughout the paper.

Fact 3.3.1. *Any norm is a convex function.*

Definition 3.3.2 (Subgradient)**.** *A subgradient of a convex function $f\colon \mathbb{R}^n \to \mathbb{R}$ at any point $\boldsymbol{x} \in \mathbb{R}^n$ is any $\boldsymbol{g} \in \mathbb{R}^n$ such that the following holds for every $\boldsymbol{y} \in \mathbb{R}^n$*

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}^\intercal(\boldsymbol{y} - \boldsymbol{x});$$

we denote by $\partial f(\boldsymbol{x})$ the set of subgradients of f at $\boldsymbol{x}$.

The following fact summarizes various useful properties of subgradients specialized to norm functions. Because we are apply norm objectives exclusively to non-negative distance vectors, we call (slightly abusing terminology) a restriction of a norm to $\mathbb{R}^n_{\geq 0}$ a norm as well.

Fact 3.3.3 ([29])**.** *Let $f\colon \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a norm and $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$. If $\boldsymbol{g}$ is a subgradient of f at $\boldsymbol{x}$, then $f(x) = \boldsymbol{g}^\intercal \boldsymbol{x}$ and $f(y) \geq \boldsymbol{g}^\intercal \boldsymbol{y}$ for all $\boldsymbol{y} \in \mathbb{R}^n_{\geq 0}$. Further, if f is monotone, there exists a subgradient $\boldsymbol{g} \in \partial f(\boldsymbol{x})$ such that $\boldsymbol{g} \geq 0$.*

The following observation is an immediate consequence of Fact 3.3.3.

Observation 3.3.4. *Let $\partial f = \bigcup_{\boldsymbol{y} \in \mathbb{R}^n_{\geq 0}} \partial f(\boldsymbol{y})$ be the set of all subgradients of f. Then for any $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$, we have that*

$$f(\boldsymbol{x}) = \max_{\boldsymbol{g} \in \partial f} \boldsymbol{g}^\intercal \boldsymbol{x}\,.$$

Definition 3.3.5 (ϵ-Approximate Subgradient)**.** *Let $f\colon \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a norm and let $\epsilon > 0$. We define the set $\partial_\epsilon f(\boldsymbol{x})$ of* ϵ-approximate subgradients of f at $\boldsymbol{x}$ *to contain all $\boldsymbol{g} \in \mathbb{R}^n_{\geq 0}$ such that the following two conditions hold*

(i) $f(\boldsymbol{y}) \geq \boldsymbol{g}^\intercal \boldsymbol{y}$ *for each $\boldsymbol{y} \in \mathbb{R}^n_{\geq 0}$, and*

(ii) $f(\boldsymbol{x}) \leq (1+\epsilon)\boldsymbol{g}^\intercal \boldsymbol{x}$.

It is known that approximate subgradients of convex functions can be computed efficiently via an (approximate) value oracle for the function through reductions shown by Grötschel, Lovasz and Schrijver in their classic book [78]. While the reduction in [78] appears to take at least $\Omega(n^{10})$ calls to the oracle, there exist faster methods assuming additional properties of the convex function, for example, see [101, 97]. Specifically for ℓ_p norms, closed formulas describing the sets of subgradients are known and used in practice.

Some Terminology and Notation Let $M = (P, F, \delta)$ be a clustering space on $n = |P|$ data points. Let $\boldsymbol{b} \in \mathbb{R}^P_{\geq 0}$ be an n-dimensional vector. We interpret $\boldsymbol{b}$ as assigning each point $p \in P$ a non-negative value denoted $b(p)$. That is, $\boldsymbol{b} = (b(p))_{p\in P}$. For example, given a subset $X \subseteq F$ of centers, we define the *distance vector* $\boldsymbol{\delta}(P, X) = (\delta(p, X))_{p\in P}$. If $B \subseteq P$ is a subset of points then $\mathbf{1}_B \in \{0,1\}^P$ denotes the *characteristic vector of B*, that is, it assigns value 1 to any $b \in B$ and 0 to any $p \in P \setminus B$. If $p \in P$ and $\alpha \geq 0$ then we denote by $\mathbf{1}_{p,\alpha}$ the binary vector $\mathbf{1}_{\mathsf{ball}(p,\alpha)\cap P}$.

3.4 ϵ-Scatter Dimension

In this section, we introduce the concept of ϵ-scatter dimension formally, which plays a central role in our algorithmic framework. The following definition is a formalization of the "center-point game" presented in the introduction.

Definition 3.4.1 (ϵ-Scatter Dimension)**.** *We are given a class $\mathcal{M}$ of finite metric spaces, a space $M = (P, F, \delta)$ in $\mathcal{M}$, and some $\epsilon \in (0,1)$. An* ϵ-scattering *in M is a sequence $(x_1, p_1) \ldots, (x_\ell, p_\ell)$ of center-point pairs $x_i \in F$, $p_i \in P$, $i \in [\ell]$ such that*

$$\begin{aligned} &\delta(x_i, p_j) \leq 1 && \text{for all } 1 \leq j < i \leq \ell && \text{(covering)} \\ &\delta(x_i, p_i) > 1+\epsilon && \text{for all } i \in [\ell] && (\epsilon\text{-refutation}) \end{aligned}$$

The ϵ-scatter dimension *of M is the maximum length of an ϵ-scattering in it. The* ϵ-scatter dimension *of $\mathcal{M}$ is the supremum of the ϵ-scatter dimension over all $M \in \mathcal{M}$.*

Note that for any ϵ-scattering $(x_1, p_1), \ldots, (x_\ell, p_\ell)$, any subsequence $(x_{i_1}, p_{i_1}), \ldots, (x_{i_{\ell'}}, p_{i_{\ell'}})$ where $i_1 < \cdots < i_{\ell'}$ and $\ell', i_j \in [\ell]$, $j \in [\ell']$ is an ϵ-scattering as well.

As described in Theorem 1.3.10, we show that bounded (algorithmic) ϵ-scatter dimension is essentially sufficient to yield an EPAS for Norm k-Clustering in the respective metric space. In Section 3.6.2 we

show that bounded treewidth and planar graph metrics, and bounded doubling metrics have bounded ϵ-scatter dimension. This allows us to obtain EPASes in all these metrics. To handle high-dimensional Euclidean space, we resort to an algorithmic version of ϵ-scatter dimension.

Optimizing the Centering Strategy Recall the example from the introduction showing that the ϵ-scatter dimension of the high-dimensional (continuous) Euclidean space $\mathbb{R}^d$ is be unbounded. We constructed an ϵ-scattering $(x_1, p_1), \ldots, (x_{d-1}, p_{d-1})$ where x_i is the i-th unit vector scaled by $1/\sqrt{2}$ and where $p_i = -x_i$ for all $i \in [d-1]$. Note that in this example the unit ball around the origin contains all the points in the sequence. Hence, the above example would collapse if the center player would improve her strategy. This motivates us to consider a variant where we replace the center player with an algorithm that computes centers more prudently. Further, employing algorithms allows us also to handle infinite spaces.

Ball Intersection Problem and Algorithm Towards this, we formalize the algorithmic problem the center player has to solve. We adopt and generalize a dual interpretation of the center-point game in which the center player is trying to find a center in the *intersection* of all unit balls around the points played by the point player. In fact, we consider the more general setting of non-uniform balls where each point p in the scattering has its own dedicated radius r.

Let $\mathcal{M}$ be a class of metric spaces (P, F, δ) with possibly infinite center sets F. We define the following search problem.

Ball Intersection
Input: A metric space $M = (P, F, \delta) \in \mathcal{M}$, a set finite set $Q \subsetneq P \times \mathbb{R}_+$ of *distance constraints*.
Output: A point $x \in F$ *satisfying* all distance constraints, that is, $\delta(x, p) \leq r$ for each $(p, r) \in Q$, if such a point exists and "fail" otherwise.

For finite metric spaces, the Ball Intersection problem can be solved efficiently by exhaustively searching the center space F. Unfortunately, we are not aware of exact algorithms for Ball Intersection for certain infinite metric spaces such as high-dimensional continuous Euclidean space. We therefore work with approximate algorithms. To define this formally, we say that a center $x \in F$ *η-satisfies* the distance constraint $(p, r) \in Q$ for some error parameter $\eta > 0$, if $\delta(x, p) \leq (1 + \eta)r$. Let $C_{\mathcal{M}}$ be a (deterministic) algorithm whose input is an instance of Ball Intersection and an error parameter $\eta > 0$. The algorithm is called an *approximate Ball Intersection algorithm* (or Ball Intersection algorithm for short) if it satisfies the following conditions.

(i) The algorithm outputs a center that η-satisfies all distance constraints or it fails.

(ii) If there exists a center satisfying all points distance constraints exactly, then the algorithm does not fail.

(iii) The running time of $C_{\mathcal{M}}$ is $\mathsf{poly}(|M|, 1/\eta)$.

We remark that there is an approximate Ball Intersection algorithm for high-dimensional Euclidean space [95], which we employ in Section 3.6.4 to prove bounded algorithmic ϵ-scatter dimension of this metric.

Algorithmic ϵ-Scatter Dimension The definition of *algorithmic ϵ-scatter dimension* is based on the notion of $(C_{\mathcal{M}}, \epsilon)$-*scattering*, which is a variant of ϵ-scattering: Centers are chosen via an (approximate) Ball Intersection algorithm $C_{\mathcal{M}}$ rather than by an adversarial center-player. Intuitively, we maintain a dynamic instance of Ball Intersection that is augmented by adding distance constraints (p, r) one by one. In the context of $(C_{\mathcal{M}}, \epsilon)$-scattering, we call the distance constraints (p, r) *requests*, which are satisfied by the Ball Intersection algorithm sequentially.

Definition 3.4.2 (Algorithmic ϵ-Scatter Dimension)**.** *Let $\mathcal{M}$ be a class of metric spaces with Ball Intersection algorithm $C_{\mathcal{M}}$, let $M = (P, F, \delta)$ be a metric in $\mathcal{M}$, and let $\epsilon \in (0, 1)$ Moreover, let $p_i \in P$, $x_i \in F$, and $r_i \in \mathbb{R}_+$ for each $i \in [\ell]$ where ℓ is a positive integer. The sequence $(x_1, p_1, r_1), \ldots, (x_\ell, p_\ell, r_\ell)$ is called an* $(C_{\mathcal{M}}, \epsilon)$-scattering *if the following two conditions hold.*

(i) We have $x_i = C_{\mathcal{M}}(M, \{(p_1, r_1), \ldots, (p_{i-1}, r_{i-1})\}, \epsilon/2)$ for each $2 \leq i \leq \ell$. (There is no requirement regarding the first center x_1 in the sequence.)

(ii) Moreover, $\delta(x_i, p_i) > (1 + \epsilon) r_i$ for each $i \in [\ell]$.

We say that $\mathcal{M}$ has algorithmic $(\epsilon, C_{\mathcal{M}})$-scatter dimension $\lambda_{\mathcal{M}}(\epsilon)$ *if any $(C_{\mathcal{M}}, \epsilon)$-scattering contains at most $\lambda_{\mathcal{M}}(\epsilon)$ many triples with the same radius value. The* algorithmic ϵ-scatter dimension *of $\mathcal{M}$ is the minimum algorithmic $(\epsilon, C_{\mathcal{M}})$-scatter dimension over any Ball Intersection algorithm $C_{\mathcal{M}}$ for $\mathcal{M}$.*

When the family $\mathcal{M}$ is clear from the context we drop the subscript $\mathcal{M}$ from $\lambda_{\mathcal{M}}(\epsilon)$ and $C_{\mathcal{M}}$. Note that, in contrast to the ϵ-scatter dimension, for algorithmic ϵ-scatter dimension we demand that the number of triples per radius value be bounded rather than the total length of the sequence. In fact, this stronger requirement would not hold for high-dimensional Euclidean spaces whereas the weaker (algorithmic) requirement turns out to be sufficient for our results. Another noteworthy difference is that a subsequence of an $(C_{\mathcal{M}}, \epsilon)$-scattering is not necessarily a $(C_{\mathcal{M}}, \epsilon)$-scattering itself because it may not be consistent with the behavior of algorithm $C_{\mathcal{M}}$.

Relation Between Algorithmic and non-Algorithmic ϵ-Scatter Dimension The following lemma shows that the algorithmic ϵ-scatter dimension indeed generalizes the ϵ-scatter dimension for finite metric spaces.

Lemma 3.4.3. *Any class of finite, explicitly given, metric spaces with ϵ-scatter dimension $\lambda(\epsilon)$ has also algorithmic ϵ-scatter dimension $\lambda(\epsilon)$.*

Proof. Let $M = (P, F, \delta)$ be a metric space in the given class along with a set Q of distance constraints. Our Ball Intersection algorithm exhaustively searches F to find a center x satisfying all distance constraints

exactly. If no such point exists the algorithm fails. Let C denote this algorithm. Consider any $(C_{\mathcal{M}}, \epsilon)$-scattering. Notice that any sub-sequence of triples with the same radius value forms an ϵ-scattering. Hence the sequence contains at most $\lambda(\epsilon)$ many triples for any radius value. □

Aspect-Ratio Lemma for Algorithmic ϵ-Scatter Dimension The following is a handy consequence of bounded algorithmic ϵ-scatter dimension that we use in proving our result. It strengthens the properties of an $(C_{\mathcal{M}}, \epsilon)$-scattering by bounding the number of triples whose radii lie in an interval of bounded aspect-ratio (rather than bounding the number of triples with the same radius value).

Lemma 3.4.4. *Let $\mathcal{M}$ be a class of metric spaces of algorithmic ϵ-scatter dimension $\lambda(\epsilon)$. Then there exists a Ball Intersection algorithm $C_{\mathcal{M}}$ with the following property. Given $\epsilon \in (0, 1)$, $a > 0$, and $\tau \geq 2$, any $(C_{\mathcal{M}}, \epsilon)$-scattering contains $O(\lambda(\epsilon/2)(\log \tau)/\epsilon)$ many triples whose radii lie in the interval $[a, \tau a]$.*

Proof. It suffices to show the weaker claim that the number of requests in the interval $[a, (1 + \epsilon/100)a]$ is at most $2\lambda(\epsilon/2)$. This claim implies the lemma because the interval $[a, \tau a]$ can be covered with $O((\log \tau)/\epsilon)$ many intervals of the form $[(1 + \epsilon/100)^j, (1 + \epsilon/100)^{j+1}]$, $j \in \mathbb{Z}$.

Let $\mathcal{A}_{\mathcal{M}}$ be an Ball Intersection algorithm such that the algorithmic $(\epsilon, \mathcal{A}_{\mathcal{M}})$-scatter dimension is $\lambda(\epsilon)$. Let $\eta \in (0, 1)$ be the input error parameter. Consider the Ball Intersection algorithm $C_{\mathcal{M}}$ that works as follows. For any of the input requests (p, r) we round r to r', which is the smallest power of $1 + \eta/50$ larger than r. We then invoke $\mathcal{A}_{\mathcal{M}}$ on the rounded requests with error parameter $\eta/2$ and output the center returned by $\mathcal{A}_{\mathcal{M}}$. Clearly, this algorithm is an $(1 + \eta)$-approximate Ball Intersection algorithm (for the original requests).

Consider any algorithmic $(C_{\mathcal{M}}, \epsilon)$-scattering $(x_1, p_1, r_1), \ldots, (x_\ell, p_\ell, r_\ell)$. Let r'_i, $i \in [\ell]$ be the rounded radii computed by $C_{\mathcal{M}}$. Let $\epsilon' = \epsilon/2$. Let $1 \leq j < i \leq \ell$. We have $\delta(x_i, p_i) > (1 + \epsilon)r_i \geq (1 + \epsilon)/(1 + \epsilon/100)r'_i \geq (1 + \epsilon')r'_i$. Moreover, we have $\delta(x_i, p_j) \leq (1 + \epsilon'/2)r'_i$. Hence, the sequence $(x_1, p_1, r'_1), \ldots, (x_\ell, p_\ell, r'_\ell)$ is an algorithmic $(C_{\mathcal{M}}, \epsilon')$-scattering. The radii in the requests (p_i, r_i), $i \in [\ell]$ that lie in the interval $[a, (1 + \epsilon/100)a]$ are rounded by $C_{\mathcal{M}}$ to at most two distinct radius values because $C_{\mathcal{M}}$ is invoked with error parameter $\eta = \epsilon/2$. Hence the (unrounded) sequence contains at most $2\lambda(\epsilon') = 2\lambda(\epsilon/2)$ many triples with radii in the interval $[a, (1 + \epsilon/100)a]$. This completes the proof of the claim and therefore of the lemma. □

3.5 Framework for Efficient Parameterized Approximation Schemes

Main Result We are now ready to state our main result. In the remainder of this section, we prove the following theorem, restated from the introduction. In Section 3.5.1, we describe the EPAS and give some intuition. In Section 3.5.2, we give a full, technical analysis.

Theorem 1.3.10. *Let $\mathcal{M}$ be a class of metric spaces that is closed under scaling distances by a positive constant. There is a randomized algorithm that computes for any Norm k-Clustering instance $\mathcal{I} = (M, f, k)$*

with metric $M = (P, F, \delta) \in \mathcal{M}$, and any $\epsilon \in (0, 1)$, with high probability a $(1+\epsilon)$-approximate solution if the following two conditions are met.

(i) There is an efficient algorithm evaluating for any distance vector $\boldsymbol{x} \in \mathbb{R}^P_{\geq 0}$ the objective $f(\boldsymbol{x})$ in time $T(f)$.

(ii) There exists a function $\lambda\colon \mathbb{R}_+ \to \mathbb{R}_+$, such that for all $\epsilon > 0$, the algorithmic ϵ-scatter dimension of $\mathcal{M}$ is at most $\lambda(\epsilon)$.

The running time of the algorithm is $\exp\left(\widetilde{O}\left(\frac{k\lambda(\epsilon/10)}{\epsilon}\right)\right) \cdot \mathsf{poly}(|M|) \cdot T(f)$.

3.5.1 Algorithm

Our algorithm is stated formally in Algorithm 3. We informally summarize the key steps of our algorithm, which we also outlined partially in the technical overview. We also give some intuition of the analysis.

Using standard enumeration techniques, we assume that we know (a sufficiently exact approximation of) the optimum objective function value OPT. Our goal is to satisfy the *convex constraint* $f(\boldsymbol{x}) \leq (1+\epsilon)\mathsf{OPT}$ imposed on the distance vector $\boldsymbol{x} \in \mathbb{R}^P_{\geq 0}$ (which represents the distance vector $\boldsymbol{\delta}(P, X)$ induced by the feasible solution $X \subseteq F$). By Observation 3.3.4, this constraint is equivalent to (infinitely many) *linear constraints* $\boldsymbol{w}^\intercal \boldsymbol{x} \leq (1+\epsilon)\mathsf{OPT}$ where $\boldsymbol{w} \in \partial f$ is any subgradient of f.

To illustrate the main idea, we first describe a highly simplified, but failed attempt. We consider in each iteration of the while-loop (lines 8–15) a candidate solution X. If $f(\boldsymbol{x}) \leq (1+\epsilon)\mathsf{OPT}$, then we are done. Otherwise, we compute an ($\epsilon/10$-approximate) subgradient $\boldsymbol{w}$ of f at $\boldsymbol{x}$ in line 9. Since $\boldsymbol{w}^\intercal \boldsymbol{x} = f(\boldsymbol{x}) > (1+\epsilon)\mathsf{OPT}$, this constitutes a *violated* linear constraint. Consider sampling a point $p \in P$ with probability proportional to its contribution $w(p)\delta(p, X)$ to the objective $f(\boldsymbol{x}) = \boldsymbol{w}^\intercal \boldsymbol{x}$ (line 11). An averaging argument shows that with probability $\Omega(\epsilon)$, the sampled point p satisfies $\delta(p, X) > (1+\epsilon/3)\delta(p, O)$ for some fixed hypothetical optimum solution O. In this event, we identified a violated *distance constraint*, and call p an $\epsilon/3$-*witness* for X. We assign p to a cluster $\kappa \in [k]$ picked uniformly at random, which equals the correct cluster of p in O with probability $1/k$. Assuming that both events occur, this allows us to add the *request* (p, r) with radius value $r = \delta(p, X)/(1+\epsilon/3)$ to the *cluster constraint* Q_κ imposed on the cluster with index κ. (See lines 13 and 14.) Here, we refer to the set Q_κ of requests for cluster κ as cluster constraint of κ.

Fix cluster index $\kappa \in [k]$. Let $(p_\kappa^{(1)}, r_\kappa^{(1)}), \ldots, (p_\kappa^{(\ell)}, r_\kappa^{(\ell)})$ be the sequence of requests added to the cluster constraint associated with cluster κ. Let $x_\kappa^{(i)}$, $i \in [\ell]$ be the center of cluster κ just before adding the request $(p_\kappa^{(i)}, r_\kappa^{(i)})$ to Q_κ. The key observation is that the sequence of triples $(x_\kappa^{(1)}, p_\kappa^{(1)}, r_\kappa^{(1)}), \ldots, (x_\kappa^{(\ell)}, p_\kappa^{(\ell)}, r_\kappa^{(\ell)})$ forms an algorithmic ϵ-scattering. We would like to argue that the length of this sequence is bounded because the algorithmic ϵ-scatter dimension is bounded. Unfortunately, the scatter dimension bounds only the number of triples per radius value but not the overall length of the sequence.

To address this issue, we compute in line 1 an initial upper bound $u(p)$ on the radius of any point $p \in P$. We (approximately) satisfy these initial distance constraints for all points in a greedy pre-processing step (see

lines 2–7). We maintain the distance constraints during the main phase by adding them as initial requests (see line 5). The upper bound $u(p)$ is a rough estimate of the smallest radius r that may be imposed on p as part of any request (p, r). We modify the sampling process in the main phase (see line 11) to sample only from a subset of points whose distance to X is not much smaller than their initial upper bound $u(p)$. We show via a careful argument that every request (p, r) we make is consistent with O with probability $\Omega(\epsilon/k)$. We argue, moreover, that all radii of requests made for a particular cluster are within a factor $O(k/\epsilon^2)$ of each other. The initial upper bounds are computed by detecting "dense" balls (line 1) in the input instance in the sense that they would receive high weight by some subgradient of the objective norm and would therefore require that any near-optimal solution must place a center in the vicinity of that dense ball.

Algorithm 3: Framework for Norm k-Clustering

Data: Instance $\mathcal{I} = ((P, F, \delta), k, f\colon \mathbb{R}^P_{\geq 0} \to \mathbb{R}_{\geq 0})$ of Norm k-Clustering, error parameter $\epsilon \in (0, 1)$, $\mathsf{OPT} > 0$, Ball Intersection algorithm C according to Lemma 3.4.4
Result: Solution X of cost at most $(1+\epsilon)\mathsf{OPT}$ if solution of cost at most OPT exists

1 For each $p \in P$, compute $u(p) = \min\{\, \alpha > 0 \mid f(\mathbf{1}_{p,\alpha/3}) \geq 3\mathsf{OPT}/\alpha \,\}$;
2 Sort P in non-decreasing order of $u(p)$;
3 Mark $p_i \in P$ if $\mathsf{ball}(p_i, u(p_i))$ is disjoint from $\mathsf{ball}(p_j, u(p_j))$ for every $j < i$;
4 Let $p^{(1)}, \ldots, p^{(k')}$ be the marked points.; `// Lemma 3.5.5 shows that` $k' \leq k$
5 Let $Q_\kappa = \{(p^{(\kappa)}, u(p^{(\kappa)}))\}$ for all $\kappa \in [k']$;
6 Let $Q_\kappa = \emptyset$ for all κ with $k' < \kappa \leq k$;
7 Let $X = (x_1, \ldots, x_k)$ be any set of centers where x_κ satisfies the requests in Q_κ;
8 **while** $f(\boldsymbol{\delta}(P, X)) > (1+\epsilon)\mathsf{OPT}$ **do**
9 $\quad \boldsymbol{w} \leftarrow \epsilon/10$-subgradient of f at $\boldsymbol{\delta}(P, X)$;
10 $\quad A \leftarrow \left\{\, p \in P \mid \delta(p, X) \geq \frac{\epsilon u(p)}{1000k} \,\right\}$;
11 $\quad$ Sample an element $p \in A$ where $\mathbb{P}p = a = \frac{w(a)\delta(a,X)}{\sum_{b\in A} w(b)\delta(b,X)}$ for any $a \in A$;
12 $\quad$ Pick cluster $\kappa \in [k]$ for p uniformly at random;
13 $\quad Q_\kappa \leftarrow Q_\kappa \cup \{(p, \delta(p, X)/(1+\epsilon/3))\}$;
14 $\quad x_\kappa \leftarrow C(Q_\kappa, \epsilon/10)$ **if** *no* x_i *was found* **then fail** ;
15 **end**
16 **return** X;

3.5.2 Analysis

Overview The analysis consists of establishing the following three facts. First, if the algorithm terminates without failure, it computes a $(1+\epsilon)$-approximation. Second, the algorithm terminates—with or without failure—after a number of iterations that depends on k and ϵ only. Third, the algorithm does not fail with a probability that depends only on k and ϵ as well.

The first step of the analysis follows immediately from the stopping criterion (line 8) of the while loop.

Observation 3.5.1 (Correctness)**.** *If the algorithm terminates successfully (that is, without failure), then it outputs a* $(1+\epsilon)$*-approximate solution.*

The second step of the analysis is summarized in the following lemma, which we prove in Subsection 3.5.2.

Lemma 3.5.2 (Runtime bound)**.** *The algorithm terminates after* $O\left(\frac{k(\log k/\epsilon)\lambda(\epsilon/10)}{\epsilon}\right)$ *iterations—with or without failure.*

With these two insights at hand, we are left with the third step summarized by the following lemma, which we prove in Subsection 3.5.2.

Lemma 3.5.3 (Probability bound)**.** *The algorithm terminates successfully (that is, without failure) with probability* $\exp\left(-\widetilde{O}\left(\frac{k\lambda(\epsilon/10)}{\epsilon}\right)\right)$.

We repeat the algorithm $\exp\left(\widetilde{O}\left(\frac{k\lambda(\epsilon/10)}{\epsilon}\right)\right)$ many times and hence succeed with high probability by Lemma 3.5.3.

The remainder of this section is dedicated to proving Lemmas 3.5.2 and 3.5.3, thereby completing proof of the main Theorem 1.3.10.

Bounding the Number of Iterations

In this subsection, we prove Lemma 3.5.2. The proof consists in three steps. First, we argue that the initial upper distance bounds $u(p)$ that we compute for each point $p \in P$ are (i) consistent with any optimum solution (Lemma 3.5.4), and (ii) approximately satisfied throughout the algorithm (Lemma 3.5.5). Second, we establish that the radii in the requests made for any particular cluster are within a bounded factor (aspect ratio) of each other (Lemma 3.5.6). The third step consists in proving that, for any particular cluster, the sequence of requests along with the corresponding centers constitute an algorithmic $(C_{\mathcal{M}}, \epsilon)$-scattering of bounded aspect ratio. Hence we can use Lemma 3.4.4 to bound the length of the sequence and thus the number of iterations by a function of k and ϵ, thereby completing the proof of Lemma 3.5.2.

Initial Upper Bounds We first show that the initial upper bounds we calculate in the algorithm are conservative in the sense that they are also respected by an optimal solution.

Lemma 3.5.4. *If O is an optimal solution then $\delta(p, O) \leq u(p)$ for any $p \in P$, where $u(p)$ is the initial upper bound computed in line 1 of Algorithm 3.*

Proof. Let $\alpha = u(p)$. For the sake of a contradiction, assume that $\delta(p, O) > \alpha$. By triangle inequality, any point $p' \in \mathsf{ball}(p, \alpha/3)$ has distance at least $2\alpha/3$ to O. Hence we have $\boldsymbol{\delta}(P, O) \geq (2\alpha/3) \cdot \mathbf{1}_{p,\alpha/3}$ and thus $f(\boldsymbol{\delta}(P, O)) \geq f((2\alpha/3) \cdot \mathbf{1}_{p,\alpha/3}) = (2\alpha/3) f(\mathbf{1}_{p,\alpha/3}) \geq (2\alpha/3) \cdot \frac{3\mathsf{OPT}}{\alpha} = 2\mathsf{OPT}$, which is a contradiction. □

The following lemma says that throughout the algorithm we approximately satisfy all upper bounds. We remark that the initialization (lines 2–7) as well as the analysis is a variant of Plesník's algorithm [109] for Priority k-Center when applied to point set P with radii $u(p)$, $p \in P$.

Lemma 3.5.5. *The number k' of points marked in line 3 in Algorithm 3 is at most k. Moreover, at any time during the execution of the while loop (lines 8–15), we have that $\delta(p, X) \leq 4u(p)$. For any request (p, r) added to some cluster constraint, we have $r \leq 4u(p)$.*

Proof. By Lemma 3.5.4 each of the balls $\mathsf{ball}(p^{(\kappa)}, u(p^{(\kappa)}))$ with marked $p^{(\kappa)}$, $\kappa \in [k']$ contains at least one point from some hypothetical optimum solution O. On the other hand, these balls are pairwise disjoint by construction. Hence $k' \leq |O| \leq k$. This also implies that the algorithm can initialize $X = (x_1, \ldots, x_k)$ in line 7 with centers satisfying all initial cluster constraints. For example, it may pick the k' centers in F closest to $p^{(\kappa)}$, $\kappa \in [k']$ and $k - k'$ many additional arbitrary centers.

Because these initial requests are never removed, they are passed to the Ball Intersection algorithm (with error parameter $\epsilon/10$; see line 14) whenever we make a change in the respective cluster. Hence, we have $\delta(p, X) \leq (1 + \epsilon/10)u(p) \leq 3u(p)/2$ for any marked point p throughout the execution of the while loop. For any point p' *not* marked, $\mathsf{ball}(p', u(p'))$ intersects $\mathsf{ball}(p, u(p))$ for some marked p. Because the points are processed in line 3 in non-decreasing order of $u(\cdot)$, we must have $u(p) \leq u(p')$. As argued before, $\mathsf{ball}(p, 3u(p)/2)$ is guaranteed to contain a center in X at any time during the while loop. This center has distance at most $u(p') + 2 \cdot 3u(p)/2 \leq 4u(p')$ from p' by triangle inequality. For the second claim, notice that $r < \delta(p, X) \leq 4u(p)$ at the time this request is processed in line 14 for the first time. □

Bounding the Aspect-Ratio of Requests The following lemma establishes that the radii of any two requests made for the same cluster are within a factor $O(k/\epsilon^2)$ from each other. The intuition is as follows. We ensure in the algorithm (see line 10) that we only sample points whose radii are within a factor $O(k/\epsilon)$ from $u(p)$. Assume that the radii, and thus the initial bounds $u(p)$, $u(p')$, in two request (p, r), (p', r') to the same cluster were very far from each other, say $r' \ll r$ and $u(p') \ll u(p)$. This would then imply that p was already (essentially) within radius r from some center before requesting (p, r) since there must be a center within radius $4u(p') \ll \epsilon r/3$ from p' by Lemma 3.5.5. This contradicts the assumption that we requested (p, r) in the first place.

Lemma 3.5.6. *Let (p, r) and (p', r') be requests added (in either order) to the same cluster constraint Q_κ, $\kappa \in [k]$ in line 13 of Algorithm 3. If $r' \leq \epsilon^2 \cdot r/(10^4 k)$ then the algorithm fails in line 14 upon making the second of the two requests.*

Proof. Assume for the sake of a contradiction that the algorithm does not fail but finds a center x_κ with $\delta(p, x_\kappa) \leq (1 + \epsilon/10)r$ and $\delta(p', x_\kappa) \leq (1 + \epsilon/10)r'$. Hence $\delta(p, p') \leq (1 + \epsilon/10)(r + r')$ by triangle inequality. By Lemma 3.5.5, we have $r \leq 4u(p)$ and $r' \leq 4u(p')$. Because we sample points from the set A defined in line 10, we have $r \geq \epsilon u(p)/(200k)$ and $r' \geq \epsilon u(p')/(200k)$.

Suppose $r' \leq \epsilon^2 r/(10^4 k)$. At the time of adding (p, r) to Q_κ the current candidate solution X satisfies

$\delta(p', X) \leq 4u(p') \leq 1000kr'/\epsilon$ by Lemma 3.5.5. Hence

$$\begin{aligned}\delta(p, X) &\leq \delta(p, p') + \delta(p', X)\\ &\leq (1 + \epsilon/10)(r + r') + 1000kr'/\epsilon\\ &\leq (1 + \epsilon/4)r\,.\end{aligned}$$

However, this is a contradiction because $\delta(p, X) = (1 + \epsilon/3)r$ when requesting (p, r) to Q_κ as can be seen from line 14. □

Leveraging Bounded Algorithmic ϵ-Scatter Dimension To complete the proof of Lemma 3.5.2, we fix some cluster and consider the sequence of triples (x, p, r) where (p, r) is a request made for this cluster and where x is the center of the cluster just before the request was made. We establish that this sequence constitutes an algorithmic $(C_{\mathcal{M}}, \epsilon)$-scattering and use Lemma 3.5.6 to bound the aspect ratio of the radii in this sequence by $O(k/\epsilon^2)$. We complete the proof via the aspect-ratio lemma 3.4.4.

Proof of Lemma 3.5.2. Fix a cluster index $\kappa \in [k]$. Let $(p_\kappa^{(1)}, r_\kappa^{(1)}), \ldots, (p_\kappa^{(\ell)}, r_\kappa^{(\ell)})$ be the sequence of requests in the order in which they are added to Q_κ in line 13. For any $i \in [\ell]$, let $x_\kappa^{(i)}$ be the center of cluster κ at the time just before requesting $(p_\kappa^{(i)}, r_\kappa^{(i)})$. Since $r_\kappa^{(i)} = \delta(p_\kappa^{(i)}, X)/(1 + \epsilon/3) \leq \delta(p_\kappa^{(i)}, x_\kappa^{(i)})/(1 + \epsilon/3)$ and since $x_\kappa^{(i)}$ is computed by invoking C on $\{(p_\kappa^{(1)}, r_\kappa^{(1)}), \ldots, (p_\kappa^{(i-1)}, r_\kappa^{(i-1)})\}$ and error parameter $\epsilon/10$, the sequence $(x_\kappa^{(1)}, p_\kappa^{(1)}, r_\kappa^{(1)}), \ldots, (x_\kappa^{(1)}, p_\kappa^{(\ell)}, r_\kappa^{(\ell)})$ is an algorithmic $\epsilon/5$-scattering.

By Lemma 3.5.6, $r_\kappa^{(i)} \in R_\kappa = \left[r_{\min}, \frac{10^4 k r_{\min}}{\epsilon^2}\right]$ for every $i \in [\ell]$ where $r_{\min}$ denotes the smallest radius in any request for cluster κ. Applying Lemma 3.4.4 to the interval R_κ, the length of the sequence is $O((\log k/\epsilon)\lambda(\epsilon/10)/\epsilon)$. Since our algorithm adds in each iteration one request to some cluster constraint, the overall number of iterations is $O(k(\log k/\epsilon)\lambda(\epsilon/10)/\epsilon)$. □

Bounding the Success Probability

The proof of Lemma 3.5.3 consists of two key steps: First, we argue that the algorithm terminates with success (that is, without failure) if the random choices made by the algorithm are "consistent" (to be defined more precisely below) with some hypothetical optimum solution. Second, we argue that consistency is maintained with sufficiently high probability in each iteration. Together with our upper bound on the number of iterations from Lemma 3.5.2, this completes the proof of the main result, Theorem 1.3.10.

Consistency Informally speaking, we mean by consistency that a fixed hypothetical solution would satisfy all current cluster constraints.

Definition 3.5.7. *Consider a fixed hypothetical optimum solution $O = (o_1, \ldots, o_k)$. We say that the current state of execution (specified by $(X, Q_1, \ldots, Q_k)$) of Algorithm 3 is* consistent with *O if for any request $(p, r) \in Q_\kappa$, $\kappa \in [k]$, we have that $\delta(p, o_\kappa) \leq r$.*

If the current state is consistent with the optimum solution O, then O certifies existence of solution to the cluster constraints $(Q_1, \ldots, Q_k)$ currently imposed. Therefore, the following observation is straightforward.

Observation 3.5.8. *If the state of the algorithm is consistent with O before executing line 14 in any iteration, then the algorithm does not fail during this iteration.*

Probability of Maintaining Consistency If the state of execution is consistent with O at the beginning of some iteration, then it remains consistent under the following two conditions. First, the point p sampled in this iteration is (randomly) assigned to the correct cluster. Second, the distance of p to the current candidate solution is sufficiently larger than its distance to O, thereby justifying the request made in line 13. This second condition motivates the following definition.

Definition 3.5.9. *Given a candidate solution X with $f(\boldsymbol{\delta}(P, X)) > (1+\epsilon)\mathsf{OPT}$, a point $p \in P$ is called an* ϵ-witness *if $\delta(p, X) > (1+\epsilon)\delta(p, O)$.*

The following lemma implies that the request made in any iteration for the sampled point is justified with probability $\Omega(\epsilon)$. It is a key part of our analysis as it links the specific way of (i) computing the initial upper bounds and (ii) sampling a witness based on these upper bounds. It is ultimately this interplay that allows us to bound the aspect ratio of the radii in the requests for a particular cluster and therefore the overall number of requests per cluster in terms of k and ϵ.

Lemma 3.5.10. *Consider a fixed iteration of the while loop of Algorithm 3 and let X be the candidate solution at the beginning of this iteration. The point sampled in line 11 is then an $\epsilon/3$-witness for X with probability $\Omega(\epsilon)$. In particular, the set A computed in line 10 is not empty.*

Proof. For any subset $S \subseteq P$ of points let $C_S = \sum_{p \in S} w(p)\delta(p, X)$ denote the *contribution of S* towards $\boldsymbol{w}^\intercal \boldsymbol{\delta}(P, X) = C_P$.

Let $W \subseteq P$ be the subset of $\epsilon/3$-witnesses of X. We claim that the contribution C_W is at least $\epsilon C_P/10$.

Suppose for the sake of a contradiction that their contribution is less. Then, using $0 < \epsilon < 1$,

$$\begin{aligned}
\mathsf{OPT} &\geq \boldsymbol{w}^\intercal \boldsymbol{\delta}(P, O) \\
&\geq \sum_{p \in P \setminus W} w(p)\delta(p, O) \\
&\geq \frac{1}{1 + \epsilon/3} \sum_{p \in P \setminus W} w(p)\delta(p, X) \\
&\geq \frac{1 - \epsilon/10}{1 + \epsilon/3} \sum_{p \in P} w(p)\delta(p, X) \\
&\geq \frac{\boldsymbol{w}^\intercal \boldsymbol{\delta}(P, X)}{1 + \epsilon/2} \\
&\geq \frac{f(\boldsymbol{\delta}(P, X))}{(1 + \epsilon/2)(1 + \epsilon/10)} \\
&\geq \frac{f(\boldsymbol{\delta}(P, X))}{1 + 3\epsilon/4}
\end{aligned}$$

which contradicts $f(\boldsymbol{\delta}(P, X)) > (1 + \epsilon)\mathsf{OPT}$.

Let $W_1, \ldots, W_k$ denote the subsets of the witnesses closest to centers $x_1, \ldots, x_k$ in X, respectively.

Let $H \subseteq [k]$ be the subset of clusters $\kappa \in [k]$ such that $C_{W_\kappa} \geq \epsilon C_P/(100k)$. Fix any cluster $\kappa \in H$. Let $\{z_1, \ldots, z_\ell\}$ be the witnesses in W_κ in non-decreasing order by the distance $\delta(z_i, x_\kappa)$, $i \in [\ell]$ to their closest cluster center x_κ. Let $j \in [\ell]$ be the minimum index j such that the contribution of the set $W_\kappa^- = \{z_1, \ldots, z_j\}$ is at least $C_{W_\kappa}/2$. This implies that also $C_{W_\kappa^+} \geq C_{W_\kappa}/2$ where $W_\kappa^+ = \{z_j, \ldots, z_\ell\}$. Hence $C_{W_\kappa^-}$ and $C_{W_\kappa^+}$ are both at least $\epsilon C_P/(200k)$ because $\kappa \in H$.

We claim that $W_\kappa^+ \subseteq A$ where A is defined as in line 10 in Algorithm 3. Towards this, let $p \in W_\kappa^+$ be arbitrary. We prove that $u(p) \leq 1000k\delta(p, x_\kappa)/\epsilon$ and hence $p \in A$. To see this, notice that $\mathsf{ball}(p, 2\delta(p, x_\kappa)) \supseteq \mathsf{ball}(x_\kappa, \delta(p, x_\kappa)) \supseteq W_\kappa^-$. On the other hand,

$$\begin{aligned}
\frac{\epsilon\mathsf{OPT}}{300k} &\leq \frac{\epsilon f(\boldsymbol{\delta}(P, X))}{300k} \\
&\leq \frac{\epsilon C_P}{200k} \\
&\leq \sum_{q \in W_\kappa^-} w(q)\delta(q, x_\kappa) \\
&\leq \delta(p, x_\kappa) \sum_{q \in W_\kappa^-} w(q)\,.
\end{aligned}$$

Setting $\alpha = 6\delta(p, x_\kappa)$, this implies that

$$\begin{aligned} f(\mathbf{1}_{p,\alpha/3}) &\geq \boldsymbol{w}^\intercal \mathbf{1}_{p,\alpha/3} \\ &= \sum_{q \in \mathsf{ball}(p,\alpha/3)} w(q) \\ &\geq \sum_{q \in W_\kappa^-} w(q) \\ &\geq \frac{\epsilon \mathsf{OPT}}{200k\delta(p, x_\kappa)} \\ &= \frac{\epsilon \cdot 3\mathsf{OPT}}{100k\alpha} \,. \end{aligned} \tag{3.1}$$

Hence $u(p) \leq 100k\alpha/\epsilon \leq 1000k\delta(p, x_\kappa)/\epsilon$ as claimed. This completes the proof of the claim that $W_\kappa^+ \subseteq A$ for any $\kappa \in H$.

As shown above, $\sum_{\kappa \in [k]} C_{W_\kappa} = C_W \geq \epsilon C_P/10$. By definition of H, we have $\sum_{\kappa \in [k] \setminus H} C_{W_\kappa} \leq \epsilon C_P/100$. Hence $\sum_{\kappa \in H} C_{W_\kappa} \geq \epsilon C_P/20$. Also, by the arguments above,

$$C_{A \cap W} \geq \sum_{\kappa \in H} C_{W_\kappa^+} \geq \sum_{\kappa \in H} \frac{C_{W_\kappa}}{2} \geq \frac{\epsilon C_P}{40} \geq \frac{\epsilon C_A}{40} \,.$$

Since we sample a point p from A with probability proportional to its contribution $C_{\{p\}}$, we sample a witness in each iteration with probability at least $\epsilon/40$.

Notice that $C_P \geq f(\boldsymbol{\delta}(P, X))/2 > 0$. The left-hand side of Equation 3.1 must therefore be positive. This implies that A is not empty. □

Overall Success Probability We are now ready to prove Lemma 3.5.3, thereby completing the proof of the main theorem 1.3.10. We establish that the state of execution is consistent before entering the while loop in Algorithm 3. The proof is completed by combining the upper bound on the number of iterations (Lemma 3.5.2) with the lower bound on the probability of maintaining consistence (Lemma 3.5.10).

Proof of Lemma 3.5.3. Let $p^{(1)}, \ldots, p^{(k')}$ be the points marked in line 3 of Algorithm 3. By Lemma 3.5.4, each $\mathsf{ball}(p^{(\kappa)}, u(p^{(\kappa)}))$, $\kappa \in [k']$ contains a point from O. By construction, these balls are moreover pairwise disjoint. Hence, by relabeling the optimum centers $O = (o_1, \ldots, o_k)$, we can assume that $\delta(p^{(\kappa)}, o_\kappa) \leq u(p^{(\kappa)})$ for each marked point p where $\kappa \in [k']$ is the index of the cluster. Therefore the state of execution of the algorithm is consistent with O just before the first execution of the while loop (lines 8–15). Assume now that the state is consistent with O at the beginning of an iteration of the while loop. By Lemma 3.5.10, we sample an $\epsilon/3$-witness p in this iteration with probability $\Omega(\epsilon)$. In this event, the request (p, r) added has radius $r = \delta(p, X)/(1 + \epsilon/3) \geq \delta(p, O)$. If additionally the cluster index $\kappa \in [k]$ picked at random is the same as the one in O—which happens with probability $\Omega(1/k)$—then the state remains consistent with O. In this event, the recomputation of the center in line 14 does not fail. By Lemma 3.5.2, the algorithm terminates after

at most $O\left(\frac{k(\log k/\epsilon)\lambda(\epsilon/10)}{\epsilon}\right)$ many iterations. Since in any iteration it does not fail with probability $\Omega(\epsilon/k)$, it succeeds overall with probability $\exp\left(-\widetilde{O}\left(\frac{k\lambda(\epsilon/10)}{\epsilon}\right)\right)$. □

3.6 ϵ-Scatter Dimension Bounds

This section is devoted to bounding the ϵ-scatter dimension in various classes of metrics, proving Theorems 1.3.6, 1.3.7, and 1.3.8 from the Introduction.

3.6.1 Bounded Doubling Dimension

In this section, we show the upper bound of the ϵ-scatter dimension of any metric space of doubling dimension d, proving Theorem 1.3.6.

Scatter Dimension and Packing Given metric (X, δ), an ϵ-packing of this metric is a subset of points $X' \subseteq X$ such that $\delta(i, j) \geq \epsilon$ for all $i, j \in X'$. This is a standard notion in the theory of metric spaces. We first observe the following connection between our ϵ-scattering and ϵ-packing.

Observation 3.6.1. *Let $(x_1, p_1), \ldots, (x_\ell, p_\ell)$ be an ϵ-scattering in a metric space (P, F, δ). Then, the set $X = \{x_2, \ldots, x_\ell\}$ fo centers is an ϵ-packing in metric $(P \cup F, \delta)$ and X is contained in a unit ball.*

Corollary 3.6.2. *The size of ϵ-packing of a unit ball in metric M is at most the ϵ-scatter dimension minus one.*

It is a well-known fact that ϵ-packing of any metric of doubling dimension d has size at most $O((1/\epsilon)^d)$. Combining this with Observation 3.6.1 yields Theorem 1.3.6.

Remark We note that the converse of Corollary 3.6.2 is false even in a very simple graph metric such as a star. In an n-node star rooted at r, a unit ball $\mathsf{ball}(r, 1)$ includes the whole graph. There exists an ϵ-packing of size $(n-1)$ by choosing the non-root nodes. However, any ϵ-scattering has length at most 2.

3.6.2 Bounded Treewidth Graphs

In this section we show that any graph of treewidth $\mathtt{tw}$ has ϵ-scatter dimension $\mathtt{tw}^{(1/\epsilon)^{O(\mathtt{tw})}}$. That is, we prove Theorem 1.3.7 for the bounded treewidth graph metric. We later show that the bound for planar graphs can be derived via an embedding result of [65]. For convenience, we abbreviate $\mathsf{ball}_{\delta_G}(r, \gamma)$ by $\mathsf{ball}_G(r, \gamma)$.

Treewidth and Spiders

Our proof relies on the notion of spiders, whose existence can serve as a "witness" to the fact that the treewidth of a graph G is high. Given an edge-weighted graph G, $X \subseteq V(G)$ and $\gamma \in (0, 1)$, a *γ-spider on X* is a set

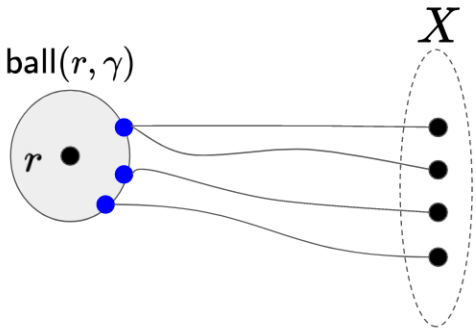


Figure 3.4: A spider $S = \mathsf{ball}_G(r, \gamma)$ on X. Paths connecting X to r are disjoint, except for nodes in S.

$S = \mathsf{ball}_G(r, \gamma)$ for some $r \in V(G)$ such that there are $|X|$ paths from S to X that are vertex-disjoint except for in S. We say that a set S is a *spider* on X if it is a γ-spider for some γ. See Figure 3.4 for illustration.

Observe that if S is a γ-spider on X, then for any $X' \subseteq X$, S is also a γ-spider on X'. The following lemma is key to our result, roughly showing that the existence of a large number of spiders implies that the treewidth of G is large.

Lemma 3.6.3. *Let G be a graph, k be an integer and $X \subseteq V(G) : |X| > 3k$. If there is a family $\mathcal{S}$ of $k+1$ pairwise disjoint spiders on X, then the treewidth of G is larger than k.*

Proof. Assume otherwise that the treewidth is at most k. Then, there exists a balanced separator $A \subseteq V(G)$ such that $|A| = k$ and a partition of $G - A = V_1 \uplus V_2$ such that $E(V_1, V_2) = \emptyset$, $|V_1 \cap X|, |V_2 \cap X| \geq |X|/3$ [51] (Folklore).

We claim that each spider $S \in \mathcal{S}$ must contain a vertex in the separator, i.e., $S \cap A \neq \emptyset$. For the sake of contradiction, say there exists $S \in \mathcal{S}$ such that $S \subseteq V_i$ for some $i \in \{1, 2\}$. Without loss of generality, let $S \subseteq V_1$. Recall that S is a spider on X. Hence there are $|X|$ many internally-vertex disjoint paths from S to distinct vertices of X. Since $|V_2 \cap X| \geq |X|/3 > k$, there are at least $k+1$ internally vertex-disjoint paths from S ($\subseteq V_1$) to $V_1 \cap X$. Since A is a (V_1, V_2)-separator, all these vertex-disjoint paths pass through A. Thus, $|A|$ is at least the number of these paths ($k+1$), which is a contradiction. We conclude that each spider $S \in \mathcal{S}$ intersects A.

Since $\mathcal{S}$ is a family of pairwise vertex-disjoint spiders, we conclude that $|A| \geq k+1$, a contradiction. □

Iteratively Finding Spiders

Our main result in this section is encapsulated in the following theorem.

Theorem 3.6.4. *If there is an ϵ-scattering of length at least $(O(k/\epsilon))^{(4/\epsilon)^{k+1}}$ in G, then graph G contains a family of $k+1$ disjoint spiders on vertex set of size greater than $3k$.*

Combining the above with Lemma 3.6.3, we can deduce that the length of any ϵ-scattering is at most $\mathtt{tw}^{(1/\epsilon)^{O(\mathtt{tw})}}$ as desired. We spend the rest of this section proving the theorem. Given ϵ-scattering σ, we say that the ϵ-packing $X = X(\sigma)$, given by Observation 3.6.1, is a canonical packing of σ.

Lemma 3.6.5. *Let σ be an ϵ-scattering of length ℓ in $G \subseteq \mathsf{ball}_G(r, 1)$ and $X = X(\sigma)$ its canonical ϵ-packing. Then, there exist*

- *a spider $S = \mathsf{ball}_G(r, \epsilon/3)$ on $X' \subseteq X : |X'| \geq c_0\epsilon \cdot (\ell/2)^{\epsilon/3}$ for some constant c_0 and*
- *a graph G' such that $S \cap V(G') = \emptyset$ and an ϵ-scattering σ' that is a subsequence of σ such that $X(\sigma') = X'$.*

Before proving this lemma, We show how this lemma implies Theorem 3.6.4. Let $G_0 = G$ contain a ϵ-scattering σ_0 of length at least $\ell_0 = (\frac{k}{c_0\epsilon})^{(4/\epsilon)^{k+1}}$ and $X_0 = X(\sigma_0)$. The lemma allows us to find a spider S_1 on X_1 of size $c_0\epsilon \cdot \ell_0^{\epsilon/3} \geq (\frac{k}{c_0\epsilon})^{(4/\epsilon)^k} = \ell_1$ for sufficiently small ϵ. Moreover, we have the graph G_1 that is disjoint with S_1 and ϵ-scattering that is a subsequence σ_1 of length ℓ_1. Since (G_1, X_1, σ_1) satisfies the preconditions of Lemma 3.6.5, we can apply it to obtain (G_2, X_2, σ_2) and so on. More formally, starting from (G_i, σ_i, X_i), we apply Lemma 3.6.5 to obtain $(G_{i+1}, \sigma_{i+1}, X_{i+1})$. We maintain the following invariant: The length of the sequence σ_i satisfies $\ell_i = |X_i| \geq (\frac{k}{c_0\epsilon})^{(4/\epsilon)^{k+1-i}}$. This allows us to find disjoint spiders $S_1, S_2, \ldots, S_{k+1}$ on $X_{k+1} : |X_{k+1}| > 3k$ as desired.

Proof of Lemma 3.6.5

Let G be contained in the unit ball $\mathsf{ball}(r, 1)$. The proof has two steps. In the first step, we find a spider S on a subset $X'' \subseteq X$ of relatively large size. In the second step, we show the graph G' obtained by removing S from G still contains a large subsequence σ' of σ whose canonical packing is a subset X' of X'' that has desired cardinality.

First step: Let T be a shortest path tree from r to X (recall that $|X| = \ell$), so vertices in X appear at the leafs of this tree. We construct an "auxiliary" tree $\widehat{T}$ on subset $\widehat{V} \subseteq V(T)$ from T inductively as follows. Let $B_r = \mathsf{ball}_T(r, \epsilon/3)$. Remove B_r from T to obtain subtrees $T_1, \ldots, T_q$ with roots $r_1, \ldots, r_q$. For each $i \in [q]$, let $X_i \subseteq X$ be the descendants of r_i in T_i that are in X. Since vertices in X are at the leaf, we have that $X_i \neq \emptyset$. We inductively perform this process on the instances $(T_1, X_q), \ldots, (T_q, X_q)$ to obtain the auxiliary subtrees $\widehat{T}_i$ for (T_i, X_i). Now create $\widehat{T}$ by connecting r to $r_1, \ldots, r_q$ in (making them direct children of r). See Figure 3.5.

For each $v \in V(\widehat{T})$, denote by T_v the subtree of T rooted at v and B_v the ball $\mathsf{ball}_{T_v}(v, \epsilon/3)$ constructed by the recursive procedure. Observe that $\bigcup_{v \in V(\widehat{T})} B_v \supseteq X$ and that the depth of $\widehat{T}$ is at most $(3/\epsilon)$ (since $\delta(r, x) \leq 1$ for all $x \in X$ and each recursion reduces the root-to-leaf distance by $\epsilon/3$.)

Claim 3.6.6. *There must be a vertex $r' \in V(\widehat{T})$ such that r' has at least $D = (\ell/2)^{\epsilon/3}$ children in $\widehat{T}$.*

Proof. Assume that the number of children is less than D for every vertex in $\widehat{T}$. Then the total number of vertices in $\widehat{T}$ is less than $2D^{3/\epsilon}$. For each such vertex $v \in V(\widehat{T})$, we have $|B_v \cap X| \leq 1$ (since X is an ϵ-packing

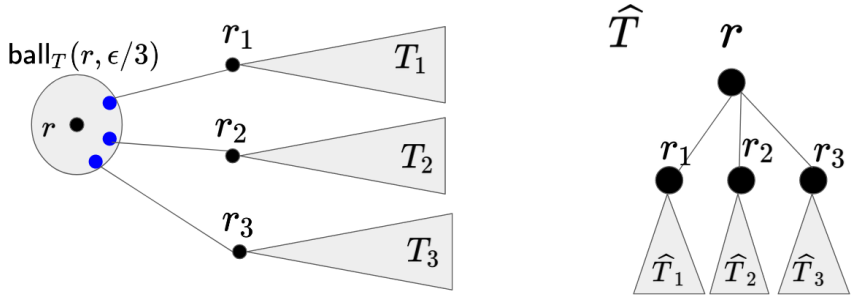


Figure 3.5: A recursive construction of tree $\widehat{T}$.

while the diameter of B_v is at most $2\epsilon/3$). Therefore, $\ell = |X| \leq \bigcup_v |B_v \cap X| \leq 2D^{3/\epsilon}$. This would imply that $D \geq (\ell/2)^{\epsilon/3}$. □

Let v be the node in $\widehat{V}$ closest to the root in $\widehat{T}$ such that there are at least D children (breaking ties arbitrarily). This means that (in the process of creating $\widehat{T}$) removing $B_v = \mathsf{ball}_{T_v}(v, \epsilon/3)$ gives us at least D subtrees $T_1, \ldots, T_D$ where each such tree contains (arbitrarily chosen) $x_i \in X_i$ as a descendant in T. Notice that S is a spider on $X'' = \{x_1, x_2, \ldots, x_D\}$.

Second step: Let σ'' be the subsequence of σ whose canonical ϵ-packing is X'', that is, $X(\sigma'') = X''$. Recall that $|X''| \geq (\ell/2)^{\epsilon/3}$. Denote the spider S by $S = \mathsf{ball}_G(s, \epsilon/3)$. In this second step, we show that $G' = G \setminus S$ still contains a long ϵ-scattering σ' which is a subsequence of σ'' such that $X' = X(\sigma') \subseteq X''$ has the desirable length.

By construction, we have that $X'' \subseteq \mathsf{ball}_G(s, 1)$. We partition vertices in X'' into at most $3/\epsilon$ subsets based on the distances to s as follows: For $i = 1, \ldots, 3/\epsilon$, let $X''_i = \{x \in X'' : \delta_G(s, x) \in (i\epsilon/3, (i+1)\epsilon/3]\}$. Define X' to be the set X''_i that has maximum cardinality, so $|X'| \geq \frac{\epsilon}{3} \cdot (\ell/2)^{\epsilon/3}$. Note that for all $u, u' \in X'$, we have $|\delta(s, u) - \delta(s, u')| < \epsilon/3$. The following claim asserts that each point in S is roughly of the same distance from every point in X' (see Figure 3.6 for illustration).

Claim 3.6.7. *Let $v \in S$ and $u, u' \in X'$. Then $|\delta(v, u) - \delta(v, u')| < \epsilon$.*

Proof. Assume w.l.o.g. that $\delta(v, u) \leq \delta(v, u')$. By the triangle inequality, $\delta(v, u') \leq \delta(v, s) + \delta(s, u') \leq \delta(v, s) + \delta(s, u) + \epsilon/3 \leq \delta(s, u) + 2\epsilon/3$. Applying triangle inequality again, we get $\delta(s, u) \leq \delta(s, v) + \delta(v, u)$, and hence the desired bound. □

The following claim will finish the proof.

Claim 3.6.8. *Let σ' be the subsequence of σ'' whose canonical packing is X'. Then σ' is a valid ϵ-scattering in $G' = G \setminus S$.*

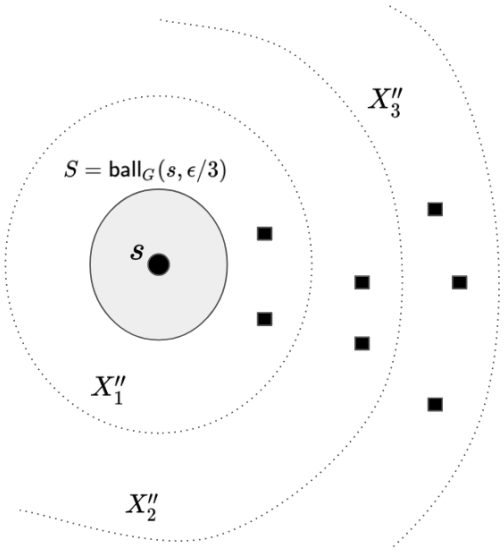


Figure 3.6: The partition of X'' into $\{X_i''\}$ based on their distance from the spider S. Rectangular points are the points in X''.

Proof. We abbreviate $\delta_{G'}$ simply by δ'. Let (x, p) and (x', p') be two pairs in σ' such that (x, p) appears before (x', p'). Since σ is scattering, we have that $\delta(x', p) \leq 1$ while $\delta(x, p), \delta(x', p'), \delta(x, p') > (1 + \epsilon)$.

Notice that the refutation properties hold for these pairs after removing S, i.e., $\delta'(x, p), \delta'(x', p')$, $\delta'(x, p') > (1 + \epsilon)$ (the distances cannot decrease after removing vertices from a graph). It suffices then to show that $\delta'(p, x') \leq 1$. To this end, we argue that any shortest path from p to x' in G cannot intersect with the ball S. Assume otherwise that there exists a shortest path Q from p to x' in G that intersects with S at some vertex $v \in S \cap Q$. Notice that $\delta(p, x') = \delta(p, v) + \delta(v, x')$. We will reach a contradiction by showing that $\delta(p, x) \leq (1 + \epsilon)$. Since $\delta(p, x) \leq \delta(p, v) + \delta(v, x)$, by Claim 3.6.8, this is at most $\delta(p, v) + \delta(v, x') + \epsilon = \delta(p, x') + \epsilon$, which would imply that $\delta(p, x) \leq (1 + \epsilon)$, contradicting to the refutation property. □

3.6.3 Bounding ϵ-Scatter Dimension via Low-Treewidth Embedding

In this section, we show a (simple) connection between bounding ϵ-scatter dimension and an active research area on embedding with additive distortion [64, 61, 37]. This connection allows us to upper bound the ϵ-scatter dimension of planar graphs.

In particular, we say that (weighted) graph class $\mathcal{G}$ admits a *t-low treewidth-diameter embedding* for function $t\colon \mathbb{N} \to \mathbb{N}$ if there exists a *deterministic* algorithm that takes G and produces a weighted graph H of treewidth at most $t(\eta)$ and an embedding $\phi\colon V(G) \to V(H)$ such that:

$$\delta_G(u, v) \leq \delta_H(\phi(u), \phi(v)) \leq \delta_G(u, v) + \eta D$$

where D is the diameter of G.

Theorem 3.6.9. *Let $\lambda_{\mathtt{tw}}(\epsilon)$ denote the the ϵ-scatter dimension of graphs of treewidth $\mathtt{tw}$ (from the previous section, this bound is at most doubly exponential in $\mathtt{tw}$). If graph class $\mathcal{G}$ admits a t-low treewidth-diameter embedding, then every metric in $\mathcal{G}$ has ϵ-scatter dimension at most $\lambda_{t(\epsilon/10)}(\epsilon/3)$.*

Proof. Let $(x_1, p_1), (x_2, p_2), \ldots, (x_\ell, p_\ell)$ be ϵ-scattering in G. Let $\eta = \epsilon/10$. Consider an embedding ϕ of G into H such that the treewidth of H is at most $t(\eta) = t(\epsilon/10)$. Notice that

- For $1 \le i < j \le \ell : \delta_H(\phi(x_j), \phi(p_i)) \le 1 + 2\eta = 1 + \epsilon/5$
- For $1 \le i \le \ell : \delta_H(\phi(x_i), \phi(p_i)) > 1 + \epsilon$

Consider (weighted) graph H' obtained by scaling the weights of H down by a factor of $(1 + \epsilon/5)$. We have that $\delta_{H'}(\phi(x_j), \phi(p_i)) \le 1$ while $\delta_{H'}(\phi(x_i), \phi(p_i)) > \frac{1+\epsilon}{1+\epsilon/5} \ge 1 + \epsilon/3$ for sufficiently small $\epsilon > 0$. This implies that the embedded sequence is $(\epsilon/3)$-scattering in H. Therefore, from Theorem 1.3.7, the length ℓ is upper bounded by $\lambda_{t(\epsilon/10)}(\epsilon/3)$. □

Now we can use the following theorem.

Theorem 3.6.10 (Theorem 1.3 of [64])**.** *There is a polynomial-time algorithm that, given an edge-weighted planar graph and given a number $\eta > 0$, outputs an embedding of the graph into a planar graph of treewidth $\mathsf{poly}(1/\eta)$ with additive error $\eta \cdot D$ where D is the diameter of the input graph.*

This implies, in our language, that planar graphs have low treewidth-diameter embedding.

Corollary 3.6.11. *Planar graphs have ϵ-scatter dimension at most* $\exp(\exp(\mathsf{poly}(1/\epsilon)))$.

3.6.4 High-Dimensional Euclidean Space

Recall, from the introduction and Sections 3.4, that the ϵ-scatter dimension of high-dimensional (continuous) Euclidean space is unbounded. In this section, we show that, in contrast, the algorithmic ϵ-scatter dimension of this metric is bounded.

Theorem 1.3.9 (Bounding Algorithmic Scatter Dimension)**.** *The continuous Euclidean space (P, F, δ), that is, $P \subsetneq \mathbb{R}^d$ finite, and $F = \mathbb{R}^d$, has algorithmic ϵ-scatter dimension $O(1/\epsilon^4 \log 1/\epsilon)$.*

We outline the proof of Theorem 1.3.9. In order to upper bound the algorithmic ϵ-scatter dimension for the continuous Euclidean space, it suffices to show that there exists an algorithm C such that the (C, ϵ)-scattering dimension in the Euclidean space is bounded. We use an algorithm by Kumar and Yildirim [95] as Ball Intersection algorithm for the high-dimensional Euclidean space. They study the Ball Intersection problem in the language of Weighted Euclidean 1-Center. They provide a Ball Intersection algorithm based on a convex optimization formulation which efficiently (and approximately) solves the Ball Intersection problem in continuous Euclidean setting for weights with bounded aspect ratio. Let C_{KY} denote this algorithm.

The following lemma is adapted from Kumar and Yildirim's work into our terminology (see Lemma 4.2 of [95]).

Lemma 3.6.12. *Given an instance (P, F, δ) of Ball Intersection in high-dimensional Euclidean space, associated radii $r(p)$ to each $p \in P$, and $\epsilon \in (0,1)$, the length of any $(C_{\mathrm{KY}}, \epsilon)$-scattering is at most $O(\tau/\epsilon^2)$ where $\tau \geq 1$ is the squared ratio of the largest radius in the requests to the smallest.*

Note that for a constant τ, Lemma 3.6.12 already gives the theorem. To complete the argument for Theorem 1.3.9 in the general setting, we show that by increasing the length of the ϵ-scattering by a multiplicative factor of $O(\log 1/\epsilon)$, we can assume that τ is $O(1/\epsilon^2)$.

Aspect-Ratio Condition The following lemma provides a sufficient condition for bounded algorithmic ϵ-scatter dimension that facilitates the design of a Ball Intersection algorithm for bounding the algorithmic ϵ-scatter dimension. In particular, this condition is key to bound the algorithmic ϵ-scatter dimension of high-dimensional continuous Euclidean spaces. It can be seen as a strenghtened converse of the aspect-ratio lemma 3.5.6 and holds for arbitrary classes of metric spaces.

Lemma 3.6.13 (Aspect-Ratio Condition)**.** *Let $\mathcal{M}$ be a class of metric spaces with Ball Intersection algorithm $C_{\mathcal{M}}$ and let $\epsilon \in (0,1)$. If any $(C_{\mathcal{M}}, \epsilon)$-scattering $(x_1, p_1, r_1), \ldots, (x_\ell, p_\ell, r_\ell)$ with $r_i \in [\epsilon/12, 1]$, $i \in [\ell]$ contains at most $\lambda(\epsilon)$ triples with the same radius, then the algorithmic ϵ-scatter dimension of $\mathcal{M}$ bounded by $O(\lambda(\epsilon) \log 1/\epsilon)$.*

To prove Lemma 3.6.13 , we assume that we are given a Ball Intersection algorithm $C_{\mathcal{M}}$ as stated. We claim that the following Ball Intersection algorithm, which invokes $C_{\mathcal{M}}$ as a sub-routine, yields algorithmic ϵ-scatter dimension $O(\lambda(\epsilon) \log 1/\epsilon)$ according to the condition of Definition 3.4.2.

Algorithm 4: Ball Intersection algorithm realizing lemma 3.6.13.

Data: Metric space $M = (P, F, \delta) \in \mathcal{M}$, requests $Q = (p_1, r_1), \ldots (p_\ell, r_\ell)$ with $p_i \in P$ and $r_i \in \mathbb{R}^+$ for $i \in [\ell]$, error parameter $\eta > 0$, Ball Intersection algorithm $C_{\mathcal{M}}$ as in Lemma 3.6.13

Result: center $x \in F$ such that $\delta(p_i, x) \leq (1+\eta) r_i$ for all $i \in [\ell]$ or "fail"

$\rho \leftarrow \min\{\, 2^{-j} \mid j \in \mathbb{N}_0 \text{ and } \min_{i \in [\ell]} r_i \leq 2^{-j} \,\}$;
$Q' \leftarrow \{\, (p_i, r_i) \mid i \in [\ell], \eta/3 \cdot r_i \leq \rho \,\}$;
$x \leftarrow C_{\mathcal{M}}(M, Q', \eta)$;
foreach $i \in [\ell]$ **do**
  if $\delta(p_i, x) > (1+\eta) r_i$ **then** "fail" ;
end
return x;

The following definition formulates a condition for two requests $(p, r), (p', r') \in Q$ under which it suffices to satisfy (p, r) in order satisfy (p', r') as well.

Definition 3.6.14. *Let $\eta \in (0,1)$ and $(p, r), (p', r') \in Q$. We say that p η-*implies *p' if $\mathsf{ball}(p, (1+\eta)r) \subseteq \mathsf{ball}(p', (1+\eta)r')$.*

Lemma 3.6.15. *Let $(p, r), (p', r') \in Q$ be two requests such that $r \leq \eta/3 \cdot r'$ for some $\eta \in (0, 1)$. If there is some center in F satisfying both requests then p η-implies p.*

Proof. Let o be the center satisfying both requests and let $x \in \mathsf{ball}(p, (1+\eta)r(p))$. By triangle inequality

$$\begin{aligned}\delta(p', x) &\leq \delta(p', o) + \delta(o, p) + \delta(p, x)\\ &\leq r' + r + (1+\eta)r\\ &\leq (1+\eta)r'\end{aligned}$$

□

Lemma 3.6.16. *Algorithm 4 is a Ball Intersection algorithm.*

Proof. Assume there is some $o \in F$ such that $\delta(p_i, o) \leq r_i$ for all $i \in [\ell]$. We want to show that our algorithm does not fail and outputs a center x such that $\delta(p_i, x) \leq (1+\eta)r_i$ for any $i \in [\ell]$.

Let Q', ρ, x be defined as in Algorithm 4. Consider any $(p_i, r_i) \in Q$. We distinguish two cases. First assume that $(p_i, r_i) \in Q'$. Assuming that $C_{\mathcal{M}}$ is a correct Ball Intersection algorithm, we have that $\delta(p_i, x) \leq (1+\eta)r_i$. On the other hand, if $(p_i, r_i) \in Q \setminus Q'$, then there is some $(p_j, r_j) \in Q'$ such that $r_j \leq \rho < \eta/3 \cdot r_i$. Hence p_j η-implies p_i by Lemma 3.6.15. As argued above, $x \in \mathsf{ball}(p_j, (1+\eta)r_j)$ and hence $x \in \mathsf{ball}(p_i, (1+\eta)r_i)$ by Definition 3.6.14. □

We conclude the proof of Lemma 3.6.13 by proving the following lemma.

Lemma 3.6.17. *Let C denote Algorithm 4. Any (C, ϵ)-scattering has then $O(\lambda(\epsilon)\log 1/\epsilon)$ triples with the same radius.*

Proof. Let Q', ρ, x be defined as in Algorithm 4. Consider an arbitrary (C, ϵ)-scattering (x_1, p_1, r_1), $\ldots, (x_\ell, p_\ell, r_\ell)$ and let Q denote the sequence of its requests, $(p_1, r_1), \ldots, (p_\ell, , r_\ell)$. Notice that, hypothetically, if we were to run the algorithm on all prefixes of Q by increasing length, the value of ρ would be monotonically decreasing over the sequence. We sub-divide the sequence into *phases*, which are maximal (contiguous) sub-sequences in which the value ρ does not change. Fix some phase. Notice that the set Q' would be inclusion-wise increasing during this phase because ρ remains unchanged. If (p_i, r_i) is an arbitrary request added to Q' at some point during the phase, then $\rho/2 \leq r_i \leq 3\rho/\eta$. Re-scaling distances by factor $\eta/(3\rho)$ and using $\eta = \epsilon/2$ shows that all requested radii during this phase lie in the range $[\epsilon/12, 1]$. Hence, by the assumption on sub-routine $C_{\mathcal{M}}$ made in Lemma 3.6.13, the scattering has at most $\lambda(\epsilon)$ many triples per radius value r_i during this phase.

We complete the proof by noting that there are at most $\log_2(12/\epsilon)$ many phases for any fixed radius value r_i, during which request with this radius are added to Q'. □

Lemmas 3.6.12 and 3.6.13 together with the observation that $\tau = (12/\epsilon)^2$ give the proof of Theorem 1.3.9.

3.7 Conclusions and Open Problems

In this chapter we provide a clean and simple EPAS that settles more than ten clustering problems (across multiple well-studied objectives as well as metric spaces) and unifies previous known EPASes. Our algorithmic framework is applicable to a wide range of well-studied objective functions in a uniform way, and unlike traditional approaches, our method does not rely on the commonly used coreset technique. A key idea behind our analysis is a new concept we introduce: the bounded ϵ-scatter dimension. Our main technical result shows that our algorithm can achieve an EPAS for any clustering objective, as long as two key conditions are met: (i)The objective function is defined by a monotone norm. (ii)The ϵ-scatter dimension of the metric space M is bounded by a function of ϵ.

There are open problems in two directions. First, can we characterize the class of metric spaces with bounded scatter dimension? For example, recently, Bourneuf and Pilipczuk [23] extended our results-which is a framework using the concept of ϵ-scatter dimension to provide Efficient Parameterized Approximation Schemes (EPASes) for Norm k-Clustering problems– to metrics induced by graphs from any fixed proper minor-closed graph class.

The second direction concerns extensions of our framework. Some clustering objectives are still missing from our framework. For instance, clustering with outliers [93, 18, 54] (in which case the cost function f is instead an anti-norm)? Even more conceptually, our current algorithm is oblivious to the structure of the input metric, but our theorem can only talk about whether an EPAS can be obtained. Is it possible for such a framework to give approximation factors in all spectrums (e.g., $(3 + o(1))$-approximation for k-Center if a general, unstructured metric space is given as input)?

Another direction is to use our framework—specifically, the concept of ϵ-scatter dimension—to design dimension-free parameterized approximation algorithms. For example, Gadekar and Inamdar [67] recently extended the idea of ϵ-scatter dimension to Hybrid k-Clustering and developed a randomized bicriteria EPAS for metrics with bounded algorithmic ϵ-scatter dimension. They showed that Hybrid k-Clustering admits a randomized bicriteria EPAS parameterized by k and ϵ in various metric spaces, including continuous Euclidean spaces of any dimension, metrics of bounded doubling dimension, bounded treewidth metrics, and metrics induced by graphs from any fixed proper minor-closed graph class[67].

Chapter 4

An $O(\log \log n)$-Approximation for Submodular Facility Location

In this chapter we present an improved approximation algorithm for SFL, and make some progress towards the open problem of achieving constant approximation for SFL.

Submodular Facility Location
Input: Instance (C, F, d) with d being a metric on $(C \cup F) \times (C \cup F) \to \mathbb{R}_{\geq 0}$, a submodular opening cost fuction function $g : 2^c \to \mathbb{R}_{\geq 0}$.
Output: An assignment where each client is assigned to a facility, that minimizes $\sum_{c \in C} d(c, \varphi(C)) + \sum_{f \in F} g(\varphi^{-1}(f))$..

Theorem 1.3.11. *There is a polynomial-time $O(\log \log n)$-approximation algorithm for SFL.*

Our approach is surprisingly simple (modulo exploiting some non-trivial results in the literature). By standard reductions (see Section 4.1.1) we can assume that $N = n + m$ is polynomial in n, hence it is sufficient to provide an $O(\log \log N)$ approximation. Our starting point is a natural (configuration) LP relaxation for the problem:

$$\begin{aligned}
\min \quad & \sum_{f \in F} \sum_{R \subseteq C} g(R) \cdot x_R^f + \sum_{c \in C} \sum_{f \in F} \sum_{R \ni c} d(c, f) \cdot x_R^f && && \text{(Conf-LP)} \\
\text{s.t.} \quad & \sum_{f \in F} \sum_{R \ni c} x_R^f = 1 && \forall c \in C; \\
& \sum_{R \subseteq C} x_R^f = 1 && \forall f \in F; \\
& x_R^f \geq 0 && \forall R \subseteq C,\ \forall f \in F.
\end{aligned}$$

In an integral solution, we interpret $x_R^f = 1$ as assigning exactly the set of clients R to the facility f. Notice

that we impose $\sum_{R\subseteq C} x^f_R = 1$. This is w.l.o.g. since $g(\emptyset) = 0$ (intuitively, $x^f_\emptyset = 1$ means that no client is assigned to f). We can solve the above LP in polynomial time.

Lemma 4.0.1. *In $\mathsf{poly}(N)$ time one can find an optimal solution to* (Conf-LP) *with $\mathsf{poly}(N)$ non-zero entries.*

Proof. Considering the dual of (Conf-LP):

$$\max\Big\{\sum_{c\in C}\alpha_c + \sum_{f\in F}\beta_f \;:\; \sum_{c\in R}\alpha_c + \beta_f \le g(R) + \sum_{c\in R} d(c,f),\ \forall R\subseteq C,\ \forall f\in F\Big\}. \tag{Conf-DLP}$$

Notice that for fixed α and β, the functions $g_f(R) \coloneqq g(R) + \sum_{c\in R} d(c,f) - \sum_{c\in R}\alpha_c - \beta_f$ are submodular. Thus, a call of a separation oracle on (Conf-DLP) is equivalent to a minimization of all functions $g_f(\cdot)$, which can be done using polynomially many oracle calls of $g(\cdot)$ [88]. Therefore, an optimal primal solution with $\mathsf{poly}(N)$ many non-zero variables for (Conf-LP) can be found in polynomial time [110, Corollary 14.1g(v)]. □

Given an optimal solution $\dot{x} = (\dot{x}^f_R)_{f\in F, R\subseteq C}$ to (Conf-LP) of cost $\mathsf{cost}(\dot{x})$ as in Lemma 4.0.1, we proceed with two main stages. In the first stage (discussed in Section 4.2) we simply sample partial assignments of clients to facilities with the distribution induced by $\dot{x}$ for $\ln\ln N$ many times. This cost at most $\ln\ln N$ times the optimal LP cost in expectation, and leads to a partial solution that covers a random subset $C_1 \subseteq C$ of clients.

In the second stage (discussed in Section 4.3) we take care of the remaining uncovered clients $C_2 = C \setminus C_1$. Let us consider the restriction $\ddot{x}$ of $\dot{x}$ to C_2. The opening cost of $\ddot{x}$ might be as large as the opening cost of $\dot{x}$. However, in expectation, the connection cost of $\ddot{x}$ is only a $1/\ln N$ fraction of the connection cost of $\dot{x}$ (as we will show).

At this point, using the probabilistic tree embedding algorithm in [58], we embed the original metric d into a (rooted) tree metric d^T over a hierarchically well-separated tree (HST) T (see Section 4.1.1 for the details). The opening cost of $\ddot{x}$ w.r.t. to the new tree instance does not change, while its connection cost grows by a factor at most $O(\log N)$ in expectation. Altogether we obtain a feasible fractional solution $\ddot{x}$ over the tree instance whose expected cost is at most $O(\mathsf{cost}(\dot{x}))$. Hence it is sufficient to develop an $O(\log\log N)$-approximate LP-rounding algorithm for the considered tree instance.

The next step is at the heart of our approach. Using the properties of HSTs and losing a constant factor in the approximation, we can further reduce our SFL tree instance to the following Descendant-Leaf Assignment problem (DLA): the facilities are leaves of T and the clients are arbitrary nodes of T. Each client c must be served by a facility contained in the subtree T_c rooted at c. The opening cost of each facility is given by $g(\cdot)$, and there are no connection costs at all. Bosman and Olver [22] describe a reduction of Submodular Joint Replenishment and Inventory Routing problems to the Nice Subadditive Cover Over Time problem. We critically observed that DLA has some similarities with the latter problem (though this connection might not be obvious at first sight, see the discussion in Section 4.1). In particular, we were able to adapt their approach to achieve the desired $O(\log\log N)$ approximation for our DLA problem.

We remark that we do not know how to get an $O(1)$ approximation for SFL on trees (even on HSTs). Though such approximation would not imply an $O(1)$ approximation for SFL with our approach (due to the first stage), finding it seems to be a natural intermediate problem to address.

The first stage of our construction might be helpful in other related problems, in particular to reduce the input problem to one on HSTs while introducing an additive $O(\log \log n)$ term in the approximation ratio.

As we discussed earlier, our basic approach is rather flexible, and it can be applied to generalizations such as, Multiplicative Opening Costs (MULTSFL), Additive Opening Costs (ADDSFL) and variant of SFL, *Universal Stochastic Facility Location* (UNIVFL).

4.1 Related Work

As mentioned earlier, Bosman and Olver [22] consider the Nice Subadditive Cover Over Time problem: roughly speaking, here we are given a set V of items and a time interval $\{1, \ldots, L\}$. Each item $v \in V$ is associated with a time window $F_v = \{s, \ldots, t\}$, $1 \leq s \leq t \leq L$. The time windows altogether have a special *left-aligned* structure whose definition we skip here. A feasible solution consists of a subset $S_t \subseteq V$ for each $t \in \{1, \ldots, L\}$, such that, for each $v \in V$, one has $v \in S_r$ for some $r \in F_v$. The goal is to minimize $\sum_{t=1}^{L} g(S_t)$, where $g(\cdot)$ is a monotone submodular set function with $g(\emptyset) = 0$. For this problem they give a $O(\log \log L)$ approximation, using a clever rounding algorithm for a convex optimization problem involving the Lovász extension of $g(\cdot)$. Intuitively, in our DLA problem (defined in Section 4.3.1) the time interval is replaced by the leaves (associated with some facility) of the tree $\tilde{T}$, and the time window of $c \in \tilde{C}$ by the set $\tilde{F}_c$. Our time windows naturally induce a laminar family, which is a special case of the left-aligned structure mentioned before. The parameter $\log L$ in their construction is replaced by the depth D of $\tilde{T}$ in our case.

In the (Metric Uncapacitated) Facility Location problem (FL) we are given a set of clients and a set of facilities in a metric space d, where each facility has an opening cost o_f. One has to select a subset of facilities $F' \subseteq F$ and assign each client c to the closest facility $F'(c)$ in F' so as to minimize $\sum_{c \in C} d(c, F'(c)) + \sum_{f \in F'} o_f$. FL is a special case of both ADDSFL and MULTSFL (and of SFL in the case of uniform opening costs). FL is among the best-studied problems in the literature from the point of view of approximation algorithms (see, e.g., [30, 102, 112]). It is known to be APX-hard [79] and the current best-known 1.488-approximation algorithm [98] is a randomized combination of the greedy JMS algorithm [89] with an LP-rounding algorithm from [26]. Lagrangian-multiplier preserving algorithms for FL are at the heart of several approximation algorithms for fundamental clustering problems, including k-MEDIAN [8, 28, 35, 38, 72, 89, 90, 99] and k-MEANS [8, 35, 76].

Various variants of FL were studied in the literature and for most of them (at least with metric connection costs) a constant approximation was eventually discovered. A notable example is the Capacitated Facility Location problem in which the number of clients that can be served from a facility is restricted by a location-specific bound. A local-search-based constant approximation for the latter problem is given in [119] (see also [9] for a more recent LP-based result). SFL is one of the most natural generalizations of (metric) FL where a

constant approximation is still not known.

Grandoni, Gupta, Leonardi, Miettinen, Sankowski, and Singh [75], among other universal stochastic problems, studied univFL in the independent activation case. However, they compare the cost of their solution with $\mathbb{E}_{A\sim\pi}[\mathsf{cost}_A(\mathsf{OPT}(A))]$, where $\mathsf{OPT}(A)$ is the optimal facility location solution restricted to clients A (while we compare with $\mathbb{E}_{A\sim\pi}[\mathsf{cost}_A(\mathsf{OPT})]$). For this setting they obtain a $O(\log n)$ approximation, which also holds for non-metric connection costs.

Gupta, Pál, Ravi, and Sinha [81] consider a 2-stage stochastic version of FL. Here in a first stage, one buys some facilities, then a subset of active clients is sampled from a given distribution. Finally, one can buy some more facilities, however at an opening cost which is increased by a multiplicative *inflation factor* σ. For this setting they present a constant approximation.

Universal stochastic problems have a natural online stochastic counterpart. For example, in the Online Stochastic Facility Location problem clients are sampled one by one, and when client c is sampled one has to connect c to an already open facility or open a new facility f and connect c to f. Garg, Gupta, Leonardi and Sankowski [68] consider this problem in the independent activation case, i.e. when the next client to be served is sampled from a probability distribution $\pi : C \to \mathbb{R}_{\geq 0}$. For this setting, they present an $O(1)$ approximation. Meyerson [105] studied a variant of the problem where an adversary chooses the set of input clients, and then a random permutation of them is presented in input (*random order model*).

We believe that it is plausible that SFL admits a constant approximation. In particular, one might consider greedy algorithms. Gupta [82] considered a natural set-cover type greedy algorithm for SFL. The same algorithm gives a 1.861-approximation when applied to the classical Facility Location problem [89]. Gupta [82, Section 2.3] showed that this algorithm produces an $\Omega(\log n)$ approximate solutions for SFL. Hence our algorithm is provably better than that one.

4.1.1 Preliminaries and Notation

We use log for the logarithm with base 2 and ln for the natural logarithm. Define $X = C \cup F$, and $N = |X| = |C \cup F|$. Given a metric d over X, we let $d_{\min}$ be the smallest non-zero distance and $d_{\max}$ be the largest distance (that we assume to be positive w.l.o.g). We use $g(c)$ as a shortcut for $g(\{c\})$.

We sometimes express a feasible solution to SFL in the form $S = (S^f)_{f\in F}$, where $S^f \subseteq C$ specifies the clients $\varphi^{-1}(f)$ assigned to f. Notice that for each $c \in C$ there is precisely one $f \in F$ with $c \in S^f$. We define a *partial assignment* as $S = (S^f)_{f\in F}$, where $S^f \subseteq C$. We say that S covers the clients $C' = \cup_{f\in F} S^f \subseteq C$. Notice that, for technical reasons, in a partial assignment we allow $S^f \cap S^{f'} \neq \emptyset$ for two distinct $f, f' \in F$ (i.e. we allow to simultaneously assign a client to more than one facility). The cost of a (partial) assignment S of the above type is defined as $\mathsf{cost}(S) := \mathrm{conn}(S) + \mathrm{open}(S)$, where $\mathrm{conn}(S) := \sum_{f\in F}\sum_{c\in S^f} d(c, f)$ is the connection cost of S and $\mathrm{open}(S) := \sum_{f\in F} g(S^f)$ is the opening cost of S. Given a (possibly infeasible) fractional solution x for (Conf-LP), we analogously define $\mathsf{cost}(x) = \mathrm{conn}(x) + \mathrm{open}(x)$, where $\mathrm{conn}(x) = \sum_{c\in C}\sum_{f\in F}\sum_{R\ni c} d(c, f)\cdot x^f_R$, and $\mathrm{open}(x) = \sum_{f\in F}\sum_{R\subseteq C} g(R)\cdot x^f_R$.

It is convenient to define the merge $S = S_1 + S_2$ of two partial assignments S_1 and S_2 naturally as follows:

(1) for each facility $f \in F$, we initially set $S^f := S_1^f \cup S_2^f$; (2) while there exist two distinct facilities f and f' with $S^f \cap S^{f'} \neq \emptyset$, replace $S^{f'}$ with $S^{f'} \setminus S^f$ (intuitively this second step guarantees that each client is assigned to no more than one facility). We observe that merging two partial assignments cannot increase the total cost.

Lemma 4.1.1. *For any two partial assignments S_1 and S_2, $\mathsf{cost}(S_1 + S_2) \leq \mathsf{cost}(S_1) + \mathsf{cost}(S_2)$.*

Proof. Let $S = S_1 + S_2$, and S' be the intermediate value of S obtained by executing only step (1) of the merge operation. One has $\mathrm{conn}(S') = \mathrm{conn}(S_1) + \mathrm{conn}(S_2)$. Furthermore, by the submodularity (hence subadditivity) of $g(\cdot)$, $\mathrm{open}(S') \leq \mathrm{open}(S_1) + \mathrm{open}(S_2)$. Clearly $\mathrm{conn}(S) \leq \mathrm{conn}(S')$, and the monotonicity of $g(\cdot)$ implies that $\mathrm{open}(S) \leq \mathrm{open}(S')$. The claim follows. □

We will exploit the following fairly standard reductions, thanks to which in the following it will be sufficient to obtain an $O(\log\log N)$ approximation for SFL. In order to distinguish between distinct instances J of the problem, we use $\mathsf{cost}_J(\varphi)$ to denote the cost of φ w.r.t. J and define similarly $\mathrm{open}_J(\varphi)$ etc.

Lemma 4.1.2. *There is a 3-approximate reduction from SFL to the special case where $m = n$.*

Proof. Let $I = (C, F, d, g(\cdot))$ be the considered instance of SFL. Consider the complete weighted graph on nodes $C \cup F$, with weights induced by d. For each client c, let $f(c)$ be the facility closest to c. We create a dummy facility $f'(c)$ and add a dummy edge $\{c, f'(c)\}$ of weight $d(c, f(c))$. Let F' be the set of newly created facilities. Observe that $|F'| = n$. Finally we remove F and consider the metric d' over $C \cup F'$ induced by the distances over the resulting graph. Let $I' = (C, F', d', g(\cdot))$ be the obtained instance of SFL. Given a solution φ' for I', we obtain a solution φ for I by simply assigning to $f(c)$ each client c' assigned to $f'(c)$ in φ'.

Let us analyze the approximation factor introduced by this reduction. We first observe that $\mathsf{cost}_I(\varphi) \leq \mathsf{cost}_{I'}(\varphi')$. Indeed, $\mathrm{open}_I(\varphi) = \mathrm{open}_{I'}(\varphi')$. Furthermore, for each each client c' assigned to $f'(c)$ by φ', the associated connection cost w.r.t. I is $d(c', f(c)) \leq d(c', c) + d(c, f(c)) = d'(c', f'(c))$. Hence $\mathrm{conn}_I(\varphi) \leq \mathrm{conn}_{I'}(\varphi')$.

Next consider an optimal solution OPT for I. For each facility f with $\mathsf{OPT}^{-1}(f) \neq \emptyset$, let $c \in \mathsf{OPT}^{-1}(f)$ be the client closest to f. We define a solution OPT' for I' by assigning all the clients in $\mathsf{OPT}^{-1}(f)$ to $f'(c)$. Again, $\mathrm{open}_I(\varphi) = \mathrm{open}_{I'}(\varphi')$. For each client c' assigned to f in OPT, its connection cost in I' is

$$d'(c', f'(c)) = d(c, c') + d(c, f(c)) \leq d(c', f) + d(c, f) + d(c, f(c)) \leq d(c', f) + 2d(c, f) \leq 3d(c', f).$$

Hence $\mathrm{conn}_{I'}(\mathsf{OPT}') \leq 3\,\mathrm{conn}_I(\mathsf{OPT})$. The claim follows. □

Lemma 4.1.3. *For any constant $\varepsilon > 0$, There is a $(1 + 4\varepsilon)$-approximate reduction from SFL to the special case where the metric d satisfies $d_{\min} = 2$ and $d_{\max} \leq \frac{2nN}{\varepsilon}$.*

Proof. Let us guess the value $L = \max_{c \in C} d(c, \mathsf{OPT}(c))$ for some optimal solution OPT. W.l.o.g. assume $L > 0$, otherwise the problem is trivial. Consider the complete weighted graph on nodes $C \cup F$ with weights

induced by d. Remove the edges of weight larger than L. We next compute a feasible solution in each connected component of the resulting graph separately. Notice that this part of the reduction is approximation preserving since no client can be assigned to a facility in a different connected component in OPT.

Let C' and F' be the clients and facilities, resp., in one such connected component G', $X' = C' \cup F'$, and d' be the metric induced by the distances in G'. Consider the corresponding SFL instance $I' = (C', F', d', g(\cdot))$. Notice that in each such instance I' one has $d'_{\max} \leq NL$. We next change the location of elements of X' as follows. We consider the ball $B(x) := \{y \in X' : d'(x,y) \leq \frac{\varepsilon}{2n}L\}$ of radius $\frac{\varepsilon}{2n}L$ around each $x \in X'$. Let $\mathcal{I}$ be a maximal (independent) set of such balls so that, if $B(x), B(y) \in \mathcal{I}$ for $x \neq y$, then $B(x) \cap B(y) = \emptyset$. For each y with $B(y) \notin \mathcal{I}$, we consider any $B(x) \in \mathcal{I}$ with $B(x) \cap B(y) \neq \emptyset$ (which must exist since $\mathcal{I}$ is maximal) and colocate y with x. Let $I'' = (C', F', d'', g(\cdot))$ be the resulting instance of SFL. Observe that $d''_{\max} \leq NL$ and $d''_{\min} \geq \frac{\varepsilon}{n}L$.

Let $\tilde{I}$ be the union of all the instances I'', and $\tilde{d}$ be the associated distances (where inter-component distances can be considered to be $+\infty$). Given a solution φ for $\tilde{I}$ (obtained by the union of all the solutions obtained for each instance I''), we return exactly the same solution φ for I.

Let us analyze the approximation factor. Notice that $\mathrm{open}_I(\varphi) = \mathrm{open}_{\tilde{I}}(\varphi)$. Furthermore, for each client c, $d(c,\varphi(c)) \leq \tilde{d}(c,\varphi(c)) + \frac{2\varepsilon}{n}L$, where in the latter term we consider the fact that each client and facility is moved at most at distance $\frac{\varepsilon}{n}L$ from the original location. Hence $\mathrm{conn}_I(\varphi) \leq \mathrm{conn}_{\tilde{I}}(\varphi) + 2\varepsilon L$. Given an optimum solution OPT for I, by a symmetric argument one has $\mathsf{cost}_{\tilde{I}}(\mathsf{OPT}) \leq \mathsf{cost}_I(\mathsf{OPT}) + 2\varepsilon L \leq (1+2\varepsilon)\mathsf{cost}_I(\mathsf{OPT})$, where we used the fact that $\mathsf{cost}_I(\mathsf{OPT}) \geq L$. Altogether an $\alpha \geq 1$ approximation algorithm for each instance I'' implies an $\alpha(1+2\varepsilon) + 2\varepsilon \leq \alpha(1+4\varepsilon)$ approximation for I.

Finally, we scale the distance d'' and $g(\cdot)$ by the same factor $\frac{2n}{\varepsilon L}$ so that $d''_{\min} = 2$ and $d''_{\max} \leq \frac{2nN}{\varepsilon}$. Clearly this final scaling is approximation preserving. □

One of the key tools that we use is the notion of probabilistic tree embedding, which we use to map the input metric into a metric on a *hierarchically well-separated tree* (HST) while stretching the distances by a small enough factor. We recall that an HST is an edge weighted rooted tree where all the leaves are at the same distance from the root r. Furthermore, on every path from a leaf to r the edge weights are $1, 2, 4, \ldots$ In particular, edges at the same level have the same weight. We will use the following construction[1] by Fakcharoenphol, Rao and Talwar [58].

Theorem 4.1.4 (FRT metric tree embedding [58])**.** *For any finite metric space (M,d) with $d_{\min} > 1$, there exists a randomized polynomial-time algorithm returning an HST T such that:*

1. *Every $a \in M$ is mapped to some leaf $map(a)$ of T (with elements at distance zero being mapped to the same leaf);*
2. *Let $d^T(a,b) := d^T(map(a), map(b))$ be the length of the path between the leaves $map(a)$ and $map(b)$ of T. Then $d^T(a,b) \geq d(a,b)$ and $\mathbb{E}\left[d^T(a,b)\right] \leq 8\log|M| \cdot d(a,b)$;*
3. *T has depth $O(\log d_{\max})$.*

[1] We slightly and trivially extend their claim to consider nodes at distance 0.

For a given set C, let $h\colon 2^C \to \mathbb{R}$ be a monotone submodular function with $h(\emptyset) = 0$. The *Lovász extension* $\hat{h}\colon [0,1]^C \to \mathbb{R}$ of $h(\cdot)$ is defined as

$$\hat{h}(y) \coloneqq \min\Big\{ \sum_{R\subseteq C} h(R)\mu_R : \sum_{R\subseteq C}\sum_{R\ni c} \mu_R = y_c \ \forall c \in C,\ \sum_{R\subseteq C} \mu_R = 1,\ \mu \geq 0 \Big\}. \tag{4.1}$$

The function $\hat{h}(\cdot)$ is convex. We remark that $\hat{h}(y)$ can be alternatively defined as

$$\hat{h}(y) \coloneqq \sum_{k=1}^{n-1} h\left(\{c_1,\ldots,c_k\}\right)\left(y_{c_k} - y_{c_{k+1}}\right) + h(C)y_{c_n}, \tag{4.2}$$

where the components of y are sorted in decreasing order, i.e. $y_{c_1} \geq y_{c_2} \geq \cdots \geq y_{c_n}$ [66, Section 6.3]. By the monotonicity of $h(\cdot)$, $\hat{h}(\cdot)$ is also non-decreasing in the sense that $\hat{h}(y) \geq \hat{h}(y')$ if $y \geq y'$.

4.2 Reducing the Connection Cost

In this section, we show how to compute a random partial assignment $S_1 = (S_1^f)_{f\in F}$ covering a random subset of clients $C_1 \coloneqq \cup_{f\in F} S_1^f \subseteq C$ with the following high-level properties: the expected cost of S_1 is "small enough" and (2) each client belongs to C_1 with "large enough" probability. In the next section, we will describe a different partial assignment $S_2 = (S_2^f)_{f\in F}$, again of small enough cost, covering the remaining clients $C_2 \coloneqq C \setminus C_1$. By merging these two partial assignments we obtain a feasible solution for the input problem of small enough total cost.

Let $\dot{x}$ be an optimal solution to (Conf-LP) with at most $\mathsf{poly}(N)$ non-zero entries that can be computed via Lemma 4.0.1. The basic idea behind the next lemma is fairly standard: we sample partial assignments according to the distribution induced by $\dot{x}$ for $\ln\ln N$ times, and merge them together.

Lemma 4.2.1. *In polynomial time one can compute a random partial assignment S_1 covering a random subset of clients C_1 such that: (1) $\mathbb{E}\left[\mathsf{cost}(S_1)\right] \leq \ln\ln(N)\cdot\mathsf{cost}(\dot{x})$ and (2) For each $c \in C$, $\mathbb{P}[c \in C_1] \geq 1 - \frac{1}{\ln N}$.*

Proof. For $i \in \{1,2,\ldots,\ln\ln N\}$ and for every $R \subseteq C$, we define a partial assignment $S(i,R)$ by setting $S^f(i,R) = R$ independently with probability $\dot{x}_R^f$ and $S^f(i,R) = \emptyset$ otherwise. Let $S_1 = \sum_{i=1}^{\ln\ln N}\sum_{R\subseteq C} S(i,R)$ be obtained by merging all these solutions, and let $C_1 = \cup_{f\in F} S_1^f$. Observe that

$$\mathbb{P}[c \notin C_1] = \prod_{f\in F}\prod_{R\ni c}(1-\dot{x}_R^f)^{\ln\ln N} \leq e^{-\ln\ln N\sum_{f\in F}\sum_{R\ni c}\dot{x}_R^f} \leq e^{-\ln\ln N} = \frac{1}{\ln N}.$$

Furthermore, by Lemma 4.1.1, $\mathbb{E}[\mathsf{cost}(S_1)]$ is upper-bounded by

$$\sum_{i=1}^{\ln\ln N}\sum_{R\subseteq C}\mathbb{E}[\mathsf{cost}(S(i,R))] = \ln\ln N\cdot\sum_{f\in F,R\subseteq C}\dot{x}_R^f\cdot\left(g(R)+\sum_{c\in R}d(c,f)\right) = \ln\ln N\cdot\mathsf{cost}(\dot{x}). \quad \square$$

Consider the partial assignment S_1 covering the random subset of clients C_1 as in the previous lemma. Let $C_2 \coloneqq C \setminus C_2$ be the remaining (uncovered) clients. Let also $\ddot{x}$ be $\dot{x}$ restricted to C_2, i.e. $\ddot{x}^f_R = \sum_{R' \subseteq C_1} \dot{x}^f_{R \cup R'}$ for $R \subseteq C_2$ and $f \in F$. The following lemma upper bounds the expected opening and connection cost of $\ddot{x}$.

Lemma 4.2.2. *One has* $\mathrm{open}(\ddot{x}) \leq \mathrm{open}(\dot{x})$ *and* $\mathbb{E}[\mathrm{conn}(\ddot{x})] \leq \frac{1}{\ln N}\mathrm{conn}(\dot{x})$.

Proof. We have $\mathrm{open}(\ddot{x}) \leq \mathrm{open}(\dot{x})$ by the monotonicity of $g(\cdot)$. For the connection cost, notice that the probability of a client c being in C_2 is at most $1/\ln N$, and only in that case one has to pay the associated connection cost. Thus by linearity of expectation, the expected connection cost of $\ddot{x}$ is at most $\mathrm{conn}(\dot{x})/\ln N$. The claim follows. □

Notice that $\ddot{x}$ is a feasible fractional solution for (Conf-LP) limited to C_2. In the following section, we show how to randomly round $\ddot{x}$ to a partial assignment S_2 which covers C_2 at expected cost $O(\log\log N) \cdot \mathsf{cost}(\ddot{x})$. It will then follow that $S_1 + S_2$ is a feasible $O(\log\log N)$-approximate solution to the input SFL instance.

4.3 Approximating SFL on an HST

Given an SFL instance and a tree embedding of $(C \cup F, d)$ into an HST T as in Theorem 4.1.4, we say that $(C \cup F, d^T, g(\cdot), map(\cdot))$ is the corresponding HST-type instance. We remark that we allow multiple clients $C(v)$ and facilities $F(v)$ to be colocated at each leaf v of T. In this section we will describe an $O(\log\log N)$-approximate LP-rounding algorithm for the considered instances w.r.t. (Conf-LP).

Lemma 4.3.1. *Given a feasible fractional solution x to* (Conf-LP) *for an HST-type SFL instance, in polynomial time one can compute a feasible (integral) solution for the same instance with cost at most* $O(\log\log N) \cdot \mathsf{cost}(x)$.

Theorem 1.3.11 directly follows.

Proof of Theorem 1.3.11. By Lemma 4.1.2 it is sufficient to describe an $O(\log\log N)$-approximation. Furthermore by Lemma 4.1.3, we can assume that $d_{\min} = 2$ and $d_{\max} \leq \frac{2nN}{\varepsilon}$.

By applying the construction of Section 4.2 we compute a random partial assignment $S_1 = (S_1^f)_{f \in F}$ covering the clients $C_1 = \cup_{f \in F} S_1^f$ with expected cost at most $O(\log\log N) \cdot \mathsf{cost}(\dot{x})$, where $\dot{x}$ is an optimal solution to (Conf-LP). Furthermore, by Lemma 4.2.2, we obtain a feasible solution $\ddot{x}$ to (Conf-LP) restricted to clients $C_2 \coloneqq C \setminus C_1$ which satisfies $\mathrm{open}(\ddot{x}) \leq \mathrm{open}(\dot{x})$ and $\mathbb{E}[\mathrm{conn}(\ddot{x})] \leq \frac{1}{\ln N}\mathrm{conn}(\dot{x})$. By applying the probabilistic tree embedding from Theorem 4.1.4 to the metric $(C_2 \cup F, d)$, we obtain an HST-type SFL instance $(C_2 \cup F, d^T, g(\cdot), map(\cdot))$ where the tree has depth $D = O(\log d_{\max}) = O(\log N)$. Observe that $\ddot{x}$ is a feasible fractional solution for (Conf-LP) restricted to C_2 on the HST-type instance. Furthermore, let $\mathrm{conn}_T(\ddot{x})$ denote the connection cost of $\ddot{x}$ w.r.t. the HST-type instance, and define similarly $\mathrm{open}_T(\ddot{x})$ and $\mathsf{cost}_T(\ddot{x})$. Then one has

$$\mathbb{E}[\mathsf{cost}_T(\ddot{x})] = \mathrm{open}(\ddot{x}) + \mathbb{E}[\mathrm{conn}_T(\ddot{x})] \leq \mathrm{open}(\dot{x}) + O(\log N) \cdot \mathbb{E}[\mathrm{conn}(\ddot{x})] \leq O(\mathsf{cost}(\dot{x})).$$

By applying the LP-rounding algorithm from Lemma 4.3.1 to $\ddot{x}$ one obtains a partial assignment $(S_2^f)_{f\in F}$ covering the clients C_2 of cost at most $O(\log\log N)\mathsf{cost}(\dot{x})$. The same solution has no larger cost in the original problem (on a non-tree metric). Altogether $S_1 + S_2$ is a feasible solution to the input SFL problem of expected cost at most $O(\log\log N)\cdot\mathsf{cost}(\dot{x}) \le O(\log\log N)\cdot\mathsf{cost}(\mathsf{OPT})$. □

In the rest of this section, we prove Lemma 4.3.1. To this aim, we will first present a reduction to a different problem that we call the Descendent-Leaf Assignment problem (DLA) (see Section 4.3.1). Then, we will present a good-enough approximation algorithm for DLA (see Section 4.3.2).

4.3.1 A Reduction to DLA

In the Descendent-Leaf Assignment problem (DLA) we are given a rooted tree $\tilde{T}$ with depth D, a set of facilities $\tilde{F}$ and a set of clients $\tilde{C}$. Each $x \in \tilde{F}\cup\tilde{C}$ is mapped to some node $v(x)$ of $\tilde{T}$, with the restriction that facilities are mapped to leaves of $\tilde{T}$. By $\tilde{F}_c$ we denote the facilities which are mapped to nodes that are descendants of $v(c)$ in T ($v(c)$ included if it is a leaf). A feasible solution consists of an assignment $\tilde{\varphi}:\tilde{C}\to\tilde{F}$ of each $c\in\tilde{C}$ to some $f\in\tilde{F}_c$. The cost of this solution is $\sum_{f\in\tilde{F}} h(\tilde{\varphi}^{-1}(f))$, where $h(\cdot)$ is a monotone submodular function over $\tilde{C}$ with $h(\emptyset)=0$. Similarly to SFL, we also express a feasible solution as $S=(S^f)_{f\in\tilde{F}}$, where $S^f=\tilde{\varphi}^{-1}(f)$, and let $\mathsf{cost}_{\mathrm{DLA}}(S)=\sum_{f\in\tilde{F}} h(S^f)$ be the associated cost. We define a convex-programming (CP) relaxation for DLA as follows:

$$\begin{aligned} \min\quad & \sum_{f\in\tilde{F}} \hat{h}(z^f) && && \text{(DLA-CP)}\\ \text{s.t.}\quad & \sum_{f\in\tilde{F}_c} z_c^f = 1 && \forall c\in\tilde{C};\\ & z_c^f \ge 0 && \forall c\in\tilde{C},\ \forall f\in\tilde{F}. \end{aligned}$$

In a 0-1 integral solution we interpret $z_c^f=1$ as c being assigned to f. Recall that $\hat{h}(\cdot)$ is convex, which makes (DLA-CP) a convex program. We also notice that each feasible assignment $S=(S^f)_{f\in\tilde{F}}$ corresponds to a feasible integral solution $z=(z^f)_{f\in\tilde{F}}$ to (DLA-CP) with $\mathsf{cost}_{\mathrm{DLA}}(S)=\mathsf{cost}_{\mathrm{DLA}}(z) \coloneqq \sum_{f\in\tilde{F}}\hat{h}(z^f)$ and vice versa. Hence indeed (DLA-CP) is a CP-relaxation of DLA.

The next lemma provides the claimed reduction from SFL on HST-type instances to DLA.

Lemma 4.3.2. *Given a polynomial-time $O(\log D)$-approximate CP-rounding algorithm for DLA w.r.t.* (DLA-CP)*, where D is the depth of the tree, there is polynomial-time $O(\log\log N)$-approximate LP-rounding algorithm for SFL on HST-type instances with tree-depth $O(\log N)$ w.r.t.* (Conf-LP).

Proof. Let $(C\cup F, d^T, g(\cdot), map(\cdot))$ be the considered HST-type instance of SFL over an HST T, and x be an input feasible fractional solution to (Conf-LP) for this instance.

We build an instance $(\tilde{C}\cup\tilde{F},\tilde{T},h(\cdot),v(\cdot))$ of DLA as follows. First, let $y_c^f \coloneqq \sum_{R\subseteq C: c\in R} x_R^f$: intuitively this is the fractional amount by which c is assigned to f in x. We set $h(\cdot)=g(\cdot)$ and $\tilde{T}=T$. Notice that

$D = O(\log N)$. We set $\tilde{F} = F$ and $v(f) = map(f)$ for each $f \in \tilde{F}$. We associate to each $c \in C$ a new client $\tilde{c} \in \tilde{C}$. Let T_v be the subtree rooted at v (containing v and all its descendants) and F_v be the facilities located in the leaves of T_v according to $map(\cdot)$. We map $\tilde{c}$ into the lowest ancestor $v(\tilde{c})$ of $map(c)$ such that $\sum_{f \in F_{v(\tilde{c})}} y_c^f \geq 1/2$. Notice that $v(\tilde{c}) = map(c)$ is possible (in which case there is at least one facility f colocated with c in T).

We next define a feasible fractional solution z for (DLA-CP) w.r.t this DLA instance as follows. For each $\tilde{c} \in \tilde{C}$ we set $z_{\tilde{c}}^f = y_c^f / (\sum_{f' \in F_{v(\tilde{c})}} y_c^{f'})$ if $f \in F_{v(\tilde{c})}$, and otherwise $z_{\tilde{c}}^f = 0$. Let $\tilde{\varphi}$ be a solution to the DLA instance obtained with the CP-rounding algorithm in the claim w.r.t. z. We obtain a feasible solution φ for the input instance by simply setting $\varphi(c) = \tilde{\varphi}(\tilde{c})$.

It remains to analyze the cost of φ. Define $\bar{z}_{\tilde{c}}^f = y_c^f / (\sum_{f' \in F_{v(\tilde{c})}} y_c^{f'})$ for all $f \in F$. Notice that $\bar{z} \geq z$. By the definition of $\hat{h}(\cdot)$ and its monotonicity, $\hat{h}(z^f) \leq \hat{h}(\bar{z}^f) = \hat{h}(y^f / (\sum_{f' \in F_{v(\tilde{c})}} y_c^{f'})) \leq 2\hat{h}(y^f) = 2\hat{g}(y^f)$. Notice that by plugging in x_R^f for μ_R in the set in (4.1) and by how y is defined w.r.t. x above, we get $\hat{g}(y^f) \leq \sum_{R \subseteq C} g(R) \cdot x_R^f$ and in particular $\sum_{f \in F} \hat{g}(y^f) \leq \text{open}(x)$. Thus, we have $\mathsf{cost}_{\text{DLA}}(z) \leq 2\,\text{open}(x)$ and

$$\text{open}(\varphi) = \mathsf{cost}_{\text{DLA}}(\tilde{\varphi}) = O(\log D) \cdot \mathsf{cost}_{\text{DLA}}(z) \leq O(\log\log N) \cdot 2\,\text{open}(x). \tag{4.3}$$

Consider next the connection cost of a given $c \in C$. If $v(\tilde{c}) = map(c)$, i.e $v(\tilde{c})$ has no child, then $d^T(c, \varphi(c)) = 0 \leq \sum_{f \in F} d^T(c, f) y_c^f$. Otherwise, let $w(c)$ be the child of $v(\tilde{c})$ along the $v(\tilde{c})$-$map(c)$ path in T. Let Δ be the weight of the edge between $v(\tilde{c})$ and $w(c)$. Observe that the distance between $v(\tilde{c})$ and the leaves in $T_{v(\tilde{c})}$ is exactly $2\Delta - 1$. Furthermore, both c and $\varphi(c)$ are located in the leaves of $T_{v(\tilde{c})}$ in the HST mapping $map(\cdot)$. Hence $d^T(c, \varphi(c)) \leq 2(2\Delta - 1)$.

By the definition of $v(\tilde{c})$, it must be the case that $\sum_{f \in F_{w(c)}} y_c^f < \frac{1}{2}$, and consequently $\sum_{f \in F \setminus F_{w(c)}} y_c^f \geq \frac{1}{2}$. For each $f \in F \setminus F_{w(c)}$, the $map(f)$-$map(c)$ path in T has length at least $2(2\Delta - 1)$. Thus

$$\sum_{f \in F} d^T(c, f) y_c^f \geq \sum_{f \in F \setminus F_{w(c)}} d^T(c, f) y_c^f \geq \frac{1}{2} 2(2\Delta - 1).$$

Therefore, the connection cost of c in φ is at most 2 times its connection cost in x. We conclude that $\text{conn}(\varphi) \leq 2\,\text{conn}(x)$. Altogether we have $\mathsf{cost}(\varphi) \leq 2\,\text{conn}(x) + O(\log\log N) \cdot 2\,\text{open}(x) \leq O(\log\log N) \cdot \mathsf{cost}(x)$. □

4.3.2 An Approximation Algorithm for DLA

In this section, we present a CP-rounding algorithm for DLA. Lemma 4.3.1 follows by chaining Lemmas 4.3.2 and 4.3.3.

Lemma 4.3.3. *Given a feasible fractional solution z to* (DLA-CP) *on an instance of DLA with tree-depth D, in polynomial time one can compute a feasible (integral) solution to the same instance of cost at most $O(\log D) \cdot \mathsf{cost}_{\text{DLA}}(z)$.*

The CP-rounding algorithm from Lemma 4.3.3 is essentially the algorithm by Bosman and Olver [22] with minor modifications that we introduced to simplify our correctness analysis. Also, the analysis of its

approximation ratio is essentially identical to [22], but we reproduce it for the sake of completeness. In particular, we will exploit the following definitions and lemma from [22]. Let $h : 2^{\tilde{C}} \to \mathbb{R}_{\geq 0}$ be a monotone submodular function with $h(\emptyset) = 0$. For a given $f \in \tilde{F}$ and a (possibly infeasible) solution z to (DLA-CP), let $L_\theta(z^f) := \{c \in \tilde{C} : z_c^f \geq \theta\}$ be the set of clients that are served fractionally by at least some value θ by f. Let also $z^{f|\theta}$ be obtained from z^f by rounding down to θ the values larger than θ, i.e. $z_c^{f|\theta} := \min\{z_c^f, \theta\}$ for each $c \in \tilde{C}$. Given $\theta \in [0,1]$ and $z^f \in [0,1]^{\tilde{C}}$, we say that the set $L_\theta(z^f)$ is α-supported (w.r.t. h) if $\hat{h}(z^f) - \hat{h}(z^{f|\theta}) \geq \alpha h(L_\theta(z^f))$.

Lemma 4.3.4 (Lemma 5.2 from [22])**.** *Given $z^f \in [0,1]^{\tilde{C}}$ and $\alpha \in (0,1]$, at least one of the following holds: (1) there exists $\theta \in [0,1]$, which can be computed in polynomial time, such that $L_\theta(z^f)$ is $\frac{\alpha}{32}$-supported; (2) $2^{1/\alpha} h(L_1(z^f)) \leq \hat{h}(z^f)$.*

Our algorithm is Algorithm 5 in the figure. Recall that $\tilde{T}_v$ is the subtree rooted at node v, where $\tilde{T}_v$ includes v and all its descendants. Furthermore, $\tilde{F}_v$ is the set of facilities mapped to the leaves of $\tilde{T}_v$. As usual the level of a node is its hop-distance from the root.

Algorithm 5:

Input: Feasible solution z to (DLA-CP)

$S^f \leftarrow \emptyset$ for all $f \in F$;
for $i = 0, \ldots, D$ **do**
  For every node v at level $D - i$, choose an arbitrary $f_v \in \tilde{F}_v$ and set $z^{f_v} \leftarrow \sum_{f' \in \tilde{F}_v} z^{f'}$ and $z^{f'} \leftarrow 0$ for all $f' \in \tilde{F}_v \setminus \{f_v\}$;
  if *there exists $\theta \in [0,1]$ such that $L_\theta(z^{f_v})$ is $\frac{1}{32\log(D+1)}$-supported* **then**
    Choose an arbitrary such θ $S^{f_v} \leftarrow S^{f_v} \cup L_\theta(z^{f_v})$ and $z_c^{f_v} \leftarrow 0$ for all $c \in L_\theta(z^{f_v})$;
  end
  else
    $S^{f_v} \leftarrow S^{f_v} \cup L_1(z^{f_v})$ and $z_c^{f_v} \leftarrow 0$ for all $c \in L_1(z^{f_v})$;
  end
end
For every $c \in \tilde{C}$, choose $f \in \tilde{F}_c$ such that $c \in S^f$ and set $S^{f'} \leftarrow S^{f'} \setminus \{c\}$ for all $f' \in \tilde{F} \setminus \{f\}$;
return $(S^f)_{f \in \tilde{F}}$

Clearly Algorithm 5 runs in polynomial time. The next two lemmas analyze the correctness and the approximation ratio of Algorithm 5, hence proving Lemma 4.3.3.

Lemma 4.3.5. *Algorithm 5 computes a feasible DLA solution.*

Proof. Consider a given client $c \in \tilde{C}$ such that $v(c)$ is at level $D - i$ in $\tilde{T}$. Let us show that the following invariant holds at the beginning of each iteration $j \leq i$: either $\sum_{f \in \tilde{F}_c} z_c^f = 1$ or $c \in S^f$ for some $f \in \tilde{F}_c$. The invariant trivially holds for $j = 0$. Assume that it holds up to the beginning of iteration $j < i$, and consider what happens during that iteration. Notice that for every node v at level $D - j > D - i$, we either have that every $f \in \tilde{F}_v$ is a descendant of $v(c)$ or every $f \in \tilde{F}_v$ is not in $\tilde{F}_c$. Therefore, in Step (3) the value of $\sum_{f \in \tilde{F}_c} z_c^f$

does not change. In more detail, it remains 1 by inductive hypothesis. The same value can decrease in Steps (5) or (8), however, this can only happen if c is added to S^{f_v} for some $f_v \in \tilde{F}_c$. Thus the invariant holds at the end of the j-th iteration, hence at the beginning of the next iteration $j+1$.

Due to the invariant, during the iteration i, when one considers the node $v = v(c)$, one has that either c already belongs to some S^f with $f \in \tilde{F}_c$, or $\sum_{f\in\tilde{F}_c} z_c^f = 1$. In the latter case, after Step (3), $z_c^{f_v} = 1$ where $f_v \in \tilde{F}_c$, so c belongs to every set $L_\theta(z^{f_v})$ with $\theta \in [0,1]$. As a consequence, c is added to S^{f_v} either in Step (5) or in Step (8).

It might happen that a client c is assigned *also* to a facility not in $\tilde{F}_c$. Step (11) guarantees that the final assignment of c is correct and unique. □

Lemma 4.3.6. *Algorithm 5 outputs a solution of cost at most $O(\log D)\cdot \mathsf{cost}_{\mathrm{DLA}}(z)$.*

Proof. Recall that $\mathsf{cost}_{\mathrm{DLA}}(z) = \sum_{f\in\tilde{F}} \hat{h}(z^f)$. We start by observing that the value of $\mathsf{cost}_{\mathrm{DLA}}(z)$ can not increase over time when z changes during the execution of the algorithm. Indeed, Steps (5) and (8) can only decrease the entries of z, hence $\mathsf{cost}_{\mathrm{DLA}}(z)$ by the monotonicity of $\hat{h}(\cdot)$. The only other changes of z happen in Step (3). Let us interpret this step as iteratively decreasing to zero $z^{f'}$ for each $f' \in \tilde{F}_v \setminus \{f_v\}$ and increasing z^{f_v} by the same amount. The decrease of the cost at each step is $\hat{h}(z^{f_v}) + \hat{h}(z^{f'}) - \hat{h}(z^{f_v}+z^{f'})$. By the alternative definition of $\hat{h}(\cdot)$ as in (4.2) and its convexity, one has $\hat{h}(z^{f_v}+z^{f'}) = 2\hat{h}\left(\frac{z^{f_v}+z^{f'}}{2}\right) \le 2\left(\frac{1}{2}\hat{h}(z^{f_v}) + \frac{1}{2}\hat{h}(z^{f'})\right) = \hat{h}(z^{f_v}) + \hat{h}(z^{f'})$. Hence the decrease of the cost is non-negative as required.

For each facility f and level i, let $\Delta_i^\theta(f)$ be the clients added to S^f in Step (5) during iteration i (possibly $\Delta_i^\theta(f) = \emptyset$). We define similarly $\Delta_i^1(f)$ w.r.t. Step (8). Notice that, by the submodularity (hence subadditivity) of $h(\cdot)$, the increase of the cost of the solution due to adding Δ to S^f is at most $h(\Delta)$. Therefore we can upper bound the cost of the final solution $S = (S^f)_{f\in\tilde{F}}$ by

$$\mathsf{cost}_{\mathrm{DLA}}(S) \coloneqq \sum_{f\in\tilde{F}} h(S^f) \le \sum_{i=0}^{D}\sum_{f\in\tilde{F}} \Big(h(\Delta_i^\theta(f)) + h(\Delta_i^1(f))\Big).$$

Let us upper bound the right-hand side of the above inequality. Let $z(i)$ denote the value of z at the beginning of iteration i. From the previous observation, we have $\hat{h}(z(i)) \le \hat{h}(z)$ for every i. By Lemma 4.3.4 with $\alpha = \frac{1}{\log(D+1)}$, for any $\Delta_i^1(f)$ one has $h(\Delta_i^1(f)) \le \frac{1}{D+1}\hat{h}(z^f(i))$. Thus

$$\sum_{i=0}^{D}\sum_{f\in\tilde{F}} h\big(\Delta_i^1(f)\big) \le \sum_{i=0}^{D}\sum_{f\in\tilde{F}} \frac{1}{D+1}\hat{h}\big(z^f(i)\big) \le \sum_{i=0}^{D}\frac{1}{D+1}\mathsf{cost}_{\mathrm{DLA}}(z(i)) \le \mathsf{cost}_{\mathrm{DLA}}(z). \tag{4.4}$$

Let $z(D+1)$ be the value of z at the end of the D-th iteration, hence in particular $\mathsf{cost}_{\mathrm{DLA}}(z(D+1)) \ge 0$. Notice that $z = z(0)$. We can lower bound $\mathsf{cost}_{\mathrm{DLA}}(z)$ by

$$\mathsf{cost}_{\mathrm{DLA}}(z) \ge \sum_{i=0}^{D}\Big(\mathsf{cost}_{\mathrm{DLA}}(z(i)) - \mathsf{cost}_{\mathrm{DLA}}(z(i+1))\Big).$$

Let $z_1(i)$ be the value of z obtained from $z(i)$ after applying Step (3) for all nodes of level $D-i$. Let also $z_2(i)$ be the value obtained from $z_1(i)$ if, for all the facilities F_i' where Step (5) is applied during iteration i, instead of setting $z_c^f = 0$ one sets $z_c^f = \theta$ for the corresponding value of θ. For the facilities not in F_i' we simply let $z_2^f(i) = z_1^f(i)$. Observe that $z(i+1) \le z_2(i) \le z_1(i) \le z(i)$. One has

$$\begin{aligned}
\mathsf{cost}_{\mathrm{DLA}}(z(i)) - \mathsf{cost}_{\mathrm{DLA}}(z(i+1)) &\ge \mathsf{cost}_{\mathrm{DLA}}(z_1(i)) - \mathsf{cost}_{\mathrm{DLA}}(z(i+1)) \\
&\ge \mathsf{cost}_{\mathrm{DLA}}(z_1(i)) - \mathsf{cost}_{\mathrm{DLA}}(z_2(i)) \\
&= \sum_{f\in\bar{F}} \hat{h}\Big(z_1^f(i)\Big) - \hat{h}\Big(z_2^f(i)\Big) = \sum_{f\in F_i'} \hat{h}\Big(z_1^f(i)\Big) - \hat{h}\Big(z_2^f(i)\Big) \\
&\ge \frac{\sum_{f\in F_i'} h\big(\Delta_i^\theta(f)\big)}{32\log(D+1)} = \frac{\sum_{f\in\bar{F}} h\big(\Delta_i^\theta(f)\big)}{32\log(D+1)}.
\end{aligned}$$

In the first two inequalities above we used the monotonicity of $\hat{h}(\cdot)$, while in the last inequality the definition of α-supported. Altogether

$$\begin{aligned}
\sum_{i=0}^{D}\sum_{f\in\bar{F}} h\Big(\Delta_i^\theta(f)\Big) &\le 32\log(D+1)\cdot\sum_{i=0}^{D}\Big(\mathsf{cost}_{\mathrm{DLA}}(z(i)) - \mathsf{cost}_{\mathrm{DLA}}(z(i+1))\Big) \\
&\le O(\log D)\cdot\mathsf{cost}_{\mathrm{DLA}}(z). \qquad (4.5)
\end{aligned}$$

By the monotonicity of $h(\cdot)$, Step (11) cannot increase the cost of the solution, hence the claim. □

4.4 Universal Stochastic Facility Location

In this section we sketch our approximation algorithm for univFL. We first present a weaker approximation factor $O(\log\log N + \log\log \frac{d_{\max}}{d_{\min}})$. Later we will show how to refine it.

Define $g(R) \coloneqq \mathbb{P}_{A\sim\pi}[R \cap A \neq \emptyset]$. We observe that this function is monotone submodular and $g(\emptyset) = 0$. Recall that $g(c) = g(\{c\})$ for every $c \in C$. W.l.o.g. we can assume $g(c) > 0$ since otherwise we can discard c. We can define the objective function of univFL for a given assignment $\varphi : C \to F$ as

$$\mathsf{cost}(\varphi) = \mathrm{conn}(\varphi) + \mathrm{open}(\varphi) = \sum_{c\in C} d(c,\varphi(c))\cdot g(c) + \sum_{f\in F} w_f \cdot g(\varphi^{-1}(f)).$$

Notice that only the connection cost changes w.r.t. multSFL. In more detail, the connection cost of each client c is scaled by the factor $g(c)$.

We can similarly define a configuration LP for univFL, and solve it by the same arguments as in Lemma 4.0.1. We next use an analogous notation as for SFL. Let $\dot{x}$ be an optimal solution to this LP with $\mathsf{poly}(N)$ many non-zero variables. We can apply the first stage of our algorithm for SFL (described in Section 4.2) with essentially no changes. This will lead to a partial assignment S_1 of expected cost $\mathbb{E}[\mathsf{cost}(S_1)] \le \ln\ln N\cdot\mathsf{cost}(\dot{x})$

and serving the clients C_1, where $\mathbb{P}[c \notin C_1] \leq \frac{1}{\ln N}$. Mapping the metric over an HST T and considering the restriction $\ddot{x}$ of $\dot{x}$ to $C_2 \coloneqq C \setminus C_1$, we obtain that $\mathbb{E}[\mathsf{cost}_{HST}(\ddot{x})] = O(\mathsf{cost}(\dot{x}))$. A reduction similar to the one in Lemma 4.3.2 works also in this case (since the scaling of the fractional solution is done on a per-client base). However in this case $D = O(\log \frac{d_{\max}}{d_{\min}})$ (since we did not reduce the ratio $\frac{d_{\max}}{d_{\min}}$ in a preprocessing step). Hence we can apply the result from Lemma 4.3.3 to obtain an assignment covering C_2 of expected cost $O(\log\log \frac{d_{\max}}{d_{\min}}) \cdot \mathsf{cost}(\dot{x})$. This concludes the sketch of the $O(\log\log N + \log\log \frac{d_{\max}}{d_{\min}})$ approximation.

We next improve this bound via a preprocessing step. Recall that $0 < \pi_{\min} \coloneqq \min_{c\in C}\{g(c)\}$. We first scale the ratio $d_{\max}/d_{\min}$. Let us guess[2] the largest distance $L = \max_{c\in C}\{d(c, \mathsf{OPT}(c))\}$ in some optimal (universal) solution OPT. Notice that $\mathsf{cost}(\mathsf{OPT}) \geq \pi_{\min} L$. We use essentially the same arguments as in Lemma 4.1.3, we can enforce that $d_{\max} \leq NL$ and $d_{\min} \geq \frac{\varepsilon}{n}\pi_{\min} L$. Hence we obtain $\frac{d_{\max}}{d_{\min}} \leq \frac{nN}{\varepsilon\pi_{\min}}$.

Now let us reduce the number of facilities m to $O(n + \log \frac{1}{\pi_{\min}})$ (hence N as well). Here we use essentially the same argument as in the proof of Lemma 4.5.1 (with $p_f = 0$). In more detail, we can assume that $m \leq 2^n$. Indeed, otherwise we can reduce the input instance to a Weighted Set Cover instance (that we can solve exactly in polynomial time) in the same way as in the mentioned lemma, with the difference that now, for $R \neq \emptyset$, we set $\kappa_R = \min_{f\in F}\{w_f \cdot g(R) + \sum_{c\in R} d(c, f) \cdot g(c)\}$. By the rest of the construction in the same lemma, we can reduce (with a constant loss in the approximation factor) our instance to one where there are $O(\log \frac{d_{\max}}{d_{\min}}) = O(\log \frac{n2^n}{\varepsilon\pi_{\min}}) = O(n + \log \frac{1}{\pi_{\min}})$ facilities per client. Altogether we reduce N to $N' = O(n(n + \log \frac{1}{\pi_{\min}}))$. Now we can apply again the above scaling trick over the distances (with N replaced by N') to obtain distances d' which satisfy:

$$\frac{d'_{\max}}{d'_{\min}} \leq \frac{nN'}{\varepsilon\pi_{\min}} = O\left(\frac{n^3 + n^2 \log \frac{1}{\pi_{\min}}}{\pi_{\min}}\right).$$

This leads to the approximation factor

$$O\left(\log\log \frac{d'_{\max}}{d'_{\min}} + \log\log N'\right) = O\left(\log\log \frac{n}{\pi_{min}}\right).$$

4.5 Generalizations of SFL

In this section we discuss some generalizations of SFL.

4.5.1 Reduction of the Number of Facilities

We consider the generalization of SFL, next called Affine SFL, where the opening cost of each facility f with assigned clients $R \neq \emptyset$ is $g_f(R) \coloneqq p_f + w_f \cdot g(R)$, where $p_f, w_f \geq 0$ are input values. Notice that this generalizes SFL with Additive (resp., Multiplicative) Opening Costs. We also observe that each $g_f(\cdot)$ is non-negative monotone submodular.

[2]Throughout this chapter, by guessing we mean trying all the (polynomially many) possible options. Each such options leads to a different solution, and we return the best one.

We show how to reduce to the case where $m = \mathsf{poly}(n)$ (hence $N = \mathsf{poly}(n)$) while loosing a constant factor in the approximation. We will use this reduction in the following sections to convert an $O(\log\log N)$ approximation into an $O(\log\log n)$ one.

Lemma 4.5.1. *For any constant $\varepsilon > 0$, there is a $(3 + 37\varepsilon)$-approximate reduction from Affine SFL to the special case where the number of facilities is $O_\varepsilon(n^3)$.*

Proof. First of all, consider the case $m \geq 2^n$. In this case we can solve the problem optimally in polynomial time via the following reduction to the Weighted Set Cover problem. For an instance $I = (C, F, d, g(\cdot))$ of Affine SFL, consider the instance $J = (\mathcal{U}, \mathcal{R}, \kappa)$ of Weighted Set Cover with universe $\mathcal{U} = C$, set collection $\mathcal{R} = 2^C$ and weight function κ given as $\kappa_R = 0$ if $R = \emptyset$ and $\kappa_R = \min_{f\in F}(p_f + w_f \cdot g(R) + \sum_{c\in R} d(c, f))$ for $R \in 2^C \setminus \{\emptyset\}$ (which can be computed in $\mathsf{poly}(N)$ time). Notice that $2^{|\mathcal{U}|} = 2^n$ which is polynomially bounded in the input size of I. The optimal solution to J induces a solution of exactly the same cost to I and vice versa. There is a simple dynamic program which solves Weighted Set Cover in time $O(2^{|U|} \cdot |U| \cdot |\mathcal{R}|)$ [63, Lemma 2]. Applying this algorithm to J, one obtains an optimal solution for the input instance I in time $O(2^n \cdot \mathsf{poly}(n, m))$, which is polynomial in m.

Hence it remains to consider the case $m \leq 2^n$. We show how to reduce the number of facilities to $O_\varepsilon(n^2 \log(nN)) = O_\varepsilon(n^3)$, while losing the approximation factor in the claim. By exactly the same reduction as in Lemma 4.1.3, we can assume that in the input metric d the maximum distance is $0 < d_{\max} \leq NL$ and the minimum non-zero distance is $d_{\min} \geq \frac{\varepsilon}{n} L$ while loosing a factor $(1 + 4\varepsilon)$ in the approximation. Here L is some value that lower bounds the cost of a given optimum solution OPT. Let us guess the largest value P of p_f over the facilities with at least one assigned client in OPT. We discard all the facilities f with $p_f > P$. Now, assuming $P > 0$, we replace each p_f with the value $p'_f := \lceil \frac{p_f \cdot n}{\varepsilon P} \rceil \cdot \frac{\varepsilon P}{n}$ ($p'_f = p_f$ for $P = 0$). Notice that this can only increase the cost of a given solution φ, however this increase is upper bounded by $n \cdot \frac{\varepsilon P}{n} \leq \varepsilon \cdot \mathsf{cost}_I(\mathsf{OPT})$, where I is the input instance of the problem. Hence this reduction preserves the approximation guarantee up to a factor $1 + \varepsilon$. After this reduction, the set $\mathcal{P}'$ of different possible values of p'_f has cardinality at most $\frac{n}{\varepsilon}$.

Let $I = (C, F, d, p', w, g(\cdot))$ be the instance of Affine SFL obtained after the above two reductions. Consider the complete edge-weighted graph on nodes $C \cup F$, with weights induced by d. We modify this graph as follows. For each client c and value $p' \in \mathcal{P}'$, we consider the set of facilities $F_{p'}$ with $p'_f = p'$. Let $F_{p'}(c, i)$, $i \geq 0$, be the facilities in $F_{p'}$ whose distances from c are in the range $[\frac{\varepsilon}{n} L \cdot (1+\varepsilon)^i, \frac{\varepsilon}{n} L \cdot (1+\varepsilon)^{i+1})$. We also define the set $F_{p'}(c, -1)$ of the facilities in $F_{p'}$ at distance 0 from c. Notice that there are at most $1+\lceil \log_{1+\varepsilon} \frac{nN}{\varepsilon} \rceil$ sets $F_{p'}(c, i)$ which are non-empty. For each $F_{p'}(c, i) \neq \emptyset$, we choose a facility $f = f_{p'}(c, i)$ with minimum value of w_f. We create a dummy facility $f' = f'_{p'}(c, i)$ with opening cost $g'_{f'}(C') = p' + w_f \cdot g(C')$ for $C' \neq \emptyset$, and add a dummy edge $\{c, f'\}$ of weight $d(c, f)$. Let F' be the set of dummy facilities. Notice that, considering also the previous reduction, one has $|F'| \leq n \cdot \frac{n}{\varepsilon} \cdot (1 + \lceil \log_{1+\varepsilon} \frac{nN}{\varepsilon} \rceil) = O(n^2 \log(nN))$. We remove the original facilities F, and let d' be the metric given by the distances in the resulting graph G' on nodes $C \cup F'$. We solve the problem on the resulting instance $I' = (C, F', d', p', w, g(\cdot))$. Given a solution φ' for I', we obtain a solution φ for I naturally as follows: if $\varphi'(c') = f'_{p'}(c, i)$, we assign c' to $f_{p'}(c, i)$.

Let us analyze the approximation factor of this final reduction. The opening costs of φ and φ' are identical. Furthermore, for each client c' assigned to $f = f_{p'}(c,i)$ in φ, and for $f' = f'_{p'}(c,i)$, one has $d(c',f) \le d(c',c) + d(c,f) = d'(c',c) + d'(c,f') = d'(c',f')$. Hence $\mathsf{cost}_I(\varphi) = \mathsf{cost}_{I'}(\varphi')$.

Next consider an optimum solution OPT for I. We construct a feasible solution OPT' for I' as follows. Let $S^f \neq \emptyset$ be the clients assigned to some $f \in F$ in OPT. Recall that the opening cost of f is $g'_f(S^f) = p'_f + w_f \cdot g(S^f)$. Let $c \in S^f$ be the client at minimum distance $d(c,f)$ from f. Define i as -1 if $d(c,f) = 0$, and otherwise, i such that $d(c,f) \in [\frac{\varepsilon}{n}L \cdot (1+\varepsilon)^i, \frac{\varepsilon}{n}L \cdot (1+\varepsilon)^{i+1})$. In OPT' we reassign all the clients in S^f to $f' = f'_{p'_f}(c,i)$. The opening cost associated with f' in OPT' is no larger than the corresponding cost in OPT since

$$p'_{f'} + w_{f'} \cdot g(S^{f'}) = p'_f + w_{f'} \cdot g(S^f) \le p'_f + w_f \cdot g(S^f).$$

In the last inequality above we used the fact that $f \in F_{p'_f}(c,i)$ and $f_{p'_f}(c,i)$ is the facility in the latter set with minimum w_f value. The connection cost of each $c' \in S^f$ w.r.t. OPT' satisfies

$$\begin{aligned} d'(c',f') &= d'(c',c) + d'(c,f') = d(c,c') + d(c, f_{p'_f}(c,i)) \\ &\le d(c',f) + d(c,f) + (1+\varepsilon)d(c,f) \le (3+\varepsilon)d(c',f). \end{aligned}$$

Altogether, $\mathsf{cost}_{I'}(\mathsf{OPT}') \le (3+\varepsilon)\mathsf{cost}_I(\mathsf{OPT})$. Considering also the first two reductions, we obtain a global reduction which preserves the approximation guarantee up to a factor $(1+4\varepsilon)(1+\varepsilon)(3+\varepsilon) \le 3 + 37\varepsilon$. □

4.5.2 SFL with Multiplicative Opening Costs

In this section we sketch the proof of the following Theorem. By Lemma 4.5.1, it is sufficient to provide an $O(\log\log N)$ approximation.

Theorem 1.3.12. *There is a polynomial-time $O(\log\log n)$-approximation algorithm for* multSFL.

For $f \in F$ and $R \subseteq C$ let $g_f(R) := w_f \cdot g(R)$. Note that $g_f(\cdot)$ is submodular, monotone and has $g(\emptyset) = 0$ for every $f \in F$. For any (partial) assignment $S = (S^f)$ and any vector $(x_R^f)_{R\subseteq C}^{f\in F}$ let also $\mathrm{open}'(S) := \sum_{f\in F} g_f(S^f)$, resp. $\mathrm{open}'(x) := \sum_{f\in F}\sum_{R\subseteq C} g_f(R)\cdot x_R^f$ and $\mathsf{cost}'(S) := \mathrm{open}'(S) + \mathrm{conn}(S)$ resp. $\mathsf{cost}'(x) := \mathrm{open}'(x) + \mathrm{conn}(x)$.

By these definitions, the LP-relaxation of the multSFL is given by the constraints from (Conf-LP) and the objective $\mathsf{cost}'(\cdot)$. In particular, the LP-relaxation of multSFL can be solved with the approach from Lemma 4.0.1. We keep the merging rule defined in Section 4.1.1 and the sampling procedure from Section 4.2. It is easy to verify that the vector $\ddot{x}$ resulting from this procedure fulfills Lemma 4.2.2 w.r.t. open' instead of open.

We reduce multSFL to a similar problem to DLA which we call DLA* which is the same problem as DLA and with the same input variables as DLA, additional inputs $\tilde{w}_f \ge 0$ for every $f \in \tilde{F}$ and cost $\mathsf{cost}^*_{\mathrm{DLA}}(\varphi) = \sum_{f\in\tilde{F}} h_f(\varphi^{-1}(f))$ where $h_f(\cdot) := \tilde{w}_f h(\cdot)$ for every $f \in \tilde{F}$. Its convex relaxation is given by the constraints in (DLA-CP) with the cost function $\mathsf{cost}^*_{\mathrm{DLA}}(z) := \sum_{f\in\tilde{F}} \hat{h}_f(z^f)$ (where $\hat{h}_f$ is the Lovász

extension of h_f). The reduction described in Lemma 4.3.2 can be reproduced to reduce MultSFL to DLA*. We define the input values of DLA* w.r.t. MultSFL in the same way we define the input values of DLA w.r.t. SFL, with additionally $\tilde{w}_f = w_f$ for every $f \in F$. Notice that $h_f(\cdot) = \tilde{w}_f h(\cdot) = g_f(\cdot) = w_f g(\cdot)$. Every reasoning made in the proof of Lemma 4.3.2 stays valid.

We now adjust Algorithm 5 for DLA* as follows: in Step 3, we select the facility $f_v \in \tilde{F}_v$ with minimum weight $\tilde{w}_{f_v}$. In the if-clause 4, we search and verify for supportedness w.r.t. h_{f_v} instead of h (which is equivalent unless $\tilde{w}_{f_v} = 0$, in which case $L_\theta(z^{f_v})$ is supported for every θ). Since the new algorithm functions exactly like Algorithm 5, except for an arbitrary selection step becoming determined (in particular, the new algorithm is a possible implementation of Algorithm 5), its correctness is implied by the correctness of Algorithm 5.

Notice that since f_v in Step 3 is now chosen to have minimal weight, we have for any $f' \in \tilde{F}_v \setminus \{f_v\}$

$$\hat{h}_{f_v}\left(z^{f_v} + z^{f'}\right) \leq \hat{h}_{f_v}\left(z^{f_v}\right) + \hat{h}_{f_v}\left(z^{f'}\right) \leq \hat{h}_{f_v}\left(z^{f_v}\right) + \hat{h}_{f'}\left(z^{f'}\right),$$

which means that the cost of z does not increase at any time by the arguments as before. Also, notice that since h_f is submodular, monotone and $h_f(\emptyset) = 0$ we can apply Lemma 4.3.4 with respect to h_{f_v} instead of h. Thus, the cost of the sets added at Step 5 and Step 8 is still bounded as in (4.4) and (4.5).

4.5.3 SFL with Additive Opening Costs

In this section we sketch the proof of the following Theorem. As in the previous section, by Lemma 4.5.1, it is sufficient to provide an $O(\log\log N)$ approximation.

Theorem 1.3.13. *There is a polynomial-time $O(\log\log n)$-approximation algorithm for* AddSFL.

Similarly to the previous section, we define the set function $g_f(\cdot)$ as $g_f(R) = g(R) + p_f$ for $R \neq \emptyset$ and $g_f(\emptyset) = 0$. As argued in the previous section, we can find an optimum to the LP relaxation of AddSFL and reduce it to the problem DLA* as defined in the last section, but with input weights $\tilde{p}_f$ instead of $\tilde{w}_f$ and $h_f(\cdot)$ as $h_f(R) \coloneqq h(R) + p_f$ for $R \neq \emptyset$, and $h_f(\emptyset) = 0$.

We adapt Algorithm 5 like in the previous section: in Step 3, we select the facility $f_v \in \tilde{F}_v$ with minimum weight $\tilde{p}_{f_v}$. In the if-clause 4, we search and verify for supportedness w.r.t. h_{f_v} instead of h. The correctness of the new algorithm here is given by the same argument as in the previous section. Notice that by (4.2) we have $\hat{h}_f(z) = \hat{h}(z) + p_f \cdot \max_{c \in \tilde{C}} z_c$, which implies $\hat{h}_{f_v}(z^{f_v} + z^{f'}) \leq \hat{h}_{f_v}(z^{f_v}) + \hat{h}_{f'}(z^{f'})$ with f_v chosen as in Step 3 in Algorithm 5. The cost of z does therefore not increase throughout the algorithm. Bounding the cost of sets added to the solution at Step 5 and Step 8 can be done, like for MultSFL, by applying Lemma 4.3.4 to h_{f_v}.

4.6 Conclusions and Open Problems

In this chapter we improved the best known approximation algorithm for Submodular Facility Location problem, we designe an $O(\log \log n)$ approximation algortihm. Our approach is rather flexible and can be easily extended to generalizations and variants of SFL. In more detail, we achieve the same approximation factor for the practically relevant generalizations of SFL where the opening cost of each facility f is of the form of $p_f + g(S^f)$ (addSFL) or $w_f \cdot g(S^f)$ (multSFL), where $p_f, w_f \geq 0$ are input values. We also obtain an improved approximation algorithm for the related Universal Stochastic Facility Location problem.

There are several possible directions for further research. The first question is whether we can achieve the same approximation ratio of $O(\log \log n)$ for the extended version of SFL where the opening costs are submodular functions of the form $g_f(S^f) = p_f + w_f \cdot g(S^f)$. We call this problem Affine SFL. It generalizes both addSFL and multSFL. The second question is whether it is possible to achieve a constant approximation algorithm for SFL over tree instance. Finally, can we achieve a constant approximation algorithm for the general SFL problem?

Bibliography

[1] Fateme Abbasi, Marek Adamczyk, Miguel Bosch-Calvo, Jarosław Byrka, Fabrizio Grandoni, Krzysztof Sornat, and Antoine Tinguely. An O(loglog n)-Approximation for Submodular Facility Location. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming (ICALP 2024)*, volume 297 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:20, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[2] Fateme Abbasi, Sandip Banerjee, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. Parameterized Approximation For Robust Clustering in Discrete Geometric Spaces. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming (ICALP 2024)*, volume 297 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[3] Fateme Abbasi, Sandip Banerjee, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. Parameterized approximation schemes for clustering with general norm objectives. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1377–1399, 2023.

[4] Mohsen Abbasi, Aditya Bhaskara, and Suresh Venkatasubramanian. Fair clustering via equitable group representations. In *Proc. ACM Conference on Fairness, Accountability, and Transparency (FAccT '21)*, pages 504–514, 2021.

[5] Mohsen Abbasi, Aditya Bhaskara, and Suresh Venkatasubramanian. Fair clustering via equitable group representations. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 504–514, 2021.

[6] Marek Adamczyk, Fabrizio Grandoni, Stefano Leonardi, and Michal Wlodarczyk. When the optimum is also blind: A new perspective on universal optimization. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages,*

and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland, volume 80 of *LIPIcs*, pages 35:1–35:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[7] Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. In *Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS'17)*, pages 61–72, 2017.

[8] Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and Euclidean k-median by primal-dual algorithms. *SIAM J. Comput.*, 49(4), 2020.

[9] Hyung-Chan An, Mohit Singh, and Ola Svensson. LP-based algorithms for capacitated facility location. *SIAM J. Comput.*, 46(1):272–306, 2017.

[10] Barbara Anthony, Vineet Goyal, Anupam Gupta, and Viswanath Nagarajan. A plant location guide for the unsure: Approximation algorithms for min-max location problems. *Mathematics of Operations Research*, 35(1):79–101, 2010.

[11] Barbara M. Anthony, Vineet Goyal, Anupam Gupta, and Viswanath Nagarajan. A plant location guide for the unsure: Approximation algorithms for min-max location problems. *Math. Oper. Res.*, 35(1):79–101, 2010.

[12] Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The hardness of approximation of euclidean k-means. In *31st International Symposium on Computational Geometry (SoCG 2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

[13] Mihai Badŏiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC'04)*, pages 250–257, 2002.

[14] Tanvi Bajpai, Deeparnab Chakrabarty, Chandra Chekuri, and Maryam Negahbani. Revisiting priority k-center: Fairness and outliers. In *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[15] Daniel Baker, Vladimir Braverman, Lingxiao Huang, Shaofeng H-C Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering in graphs of bounded treewidth. In *International Conference on Machine Learning (ICML'20)*, pages 569–579. PMLR, 2020.

[16] Daniel Baker, Vladimir Braverman, Lingxiao Huang, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering in graphs of bounded treewidth. In *Proc. 37th International Conference on Machine Learning (ICML'20)*, volume 119, pages 569–579, 2020.

[17] Sayan Bandyapadhyay, Zachary Friggstad, and Ramin Mousavi. Parameterized approximation algorithms for k-center clustering and variants. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3895–3903, 2022.

[18] Sandip Banerjee, Rafail Ostrovsky, and Yuval Rabani. Min-sum clustering (with outliers). In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, (APPROX/RANDOM'21)*, volume 207 of *LIPIcs*, pages 16:1–16:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[19] Nikhil Bansal, Rohit Khandekar, Jochen Könemann, Viswanath Nagarajan, and Britta Peis. On generalizations of network design problems with degree bounds. *Mathematical Programming*, 141(1):479–506, 2013.

[20] Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Faster algorithms for the constrained k-means problem. *Theory of computing systems*, 62(1):93–115, 2018.

[21] Sayan Bhattacharya, Parinya Chalermsook, Kurt Mehlhorn, and Adrian Neumann. New approximability results for the robust k-median problem. In *Proc. Scandinavian Workshop on Algorithm Theory (SWAT'14)*, pages 50–61, 2014.

[22] Thomas Bosman and Neil Olver. Improved approximation algorithms for inventory problems. In Daniel Bienstock and Giacomo Zambelli, editors, *Integer Programming and Combinatorial Optimization - 21st International Conference, IPCO 2020, London, UK, June 8-10, 2020*, volume 12125 of *Lecture Notes in Computer Science*, pages 91–103. Springer, 2020.

[23] Romain Bourneuf and Marcin Pilipczuk. Bounding ε-scatter dimension via metric sparsity. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3155–3171. SIAM, 2025.

[24] Vladimir Braverman, Shaofeng H-C Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for ordered weighted clustering. In *Proc. International Conference on Machine Learning (ICML'19)*, pages 744–753, 2019.

[25] Vladimir Braverman, Shaofeng H-C Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering in excluded-minor graphs and beyond. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA'21)*, pages 2679–2696. SIAM, 2021.

[26] Jarosław Byrka and Karen Aardal. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. *SIAM J. Comput.*, 39(6):2212–2231, 2010.

[27] Jarosław Byrka, Krzysztof Sornat, and Joachim Spoerhase. Constant-factor approximation for ordered k median. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 620–631, 2018.

[28] T.W. Byrka, J.and Pensyl, B. Rybicki, A. Srinivasan, and K. Trinh. An improved approximation algorithm for k-median and positive correlation in budgeted optimization. *ACM Trans. Algorithms*, 13(2)(23):1–31, 2013.

[29] Deeparnab Chakrabarty and Chaitanya Swamy. Approximation algorithms for minimum norm and ordered optimization problems. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 126–137, 2019.

[30] Moses Charikar and Sudipto Guha. Improved combinatorial algorithms for facility location problems. *SIAM J. Comput.*, 34(4):803–824, 2005.

[31] Eden Chlamtáč, Yury Makarychev, and Ali Vakilian. Approximating fair clustering with cascaded norm objectives. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'22)*, pages 2664–2683. SIAM, 2022.

[32] Vincent Cohen-Addad. A fast approximation scheme for low-dimensional k-means. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 430–440. SIAM, 2018.

[33] Vincent Cohen-Addad, Hossein Esfandiari, Vahab Mirrokni, and Shyam Narayanan. Improved approximations for euclidean k-means and k-median, via nested quasi-independent sets. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1621–1628, 2022.

[34] Vincent Cohen-Addad, Hossein Esfandiari, Vahab S. Mirrokni, and Shyam Narayanan. Improved approximations for Euclidean k-means and k-median, via nested quasi-independent sets. In *Proc. 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC'22)*, pages 1621–1628, 2022.

[35] Vincent Cohen-Addad, Hossein Esfandiari, Vahab S. Mirrokni, and Shyam Narayanan. Improved approximations for Euclidean k-means and k-median, via nested quasi-independent sets. In Stefano Leonardi and Anupam Gupta, editors, *54th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2022, Rome, Italy, June 20 - 24, 2022*, pages 1621–1628. ACM, 2022.

[36] Vincent Cohen-Addad, Andreas Emil Feldmann, and David Saulpic. Near-linear time approximation schemes for clustering in doubling metrics. *Journal of the ACM (JACM)*, 68(6):1–34, 2021.

[37] Vincent Cohen-Addad, Arnold Filtser, Philip N Klein, and Hung Le. On light spanners, low-treewidth embeddings and efficient traversing in minor-free graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS'20)*, pages 589–600. IEEE, 2020.

[38] Vincent Cohen-Addad, Fabrizio Grandoni, Euiwoong Lee, and Chris Schwiegelshohn. Breaching the 2 LMP approximation barrier for facility location with applications to k-median. In Nikhil Bansal and Viswanath Nagarajan, editors, *34th ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 940–986. SIAM, 2023.

[39] Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. Tight fpt approximations for k-median and k-means. *arXiv preprint arXiv:1904.12334*, 2019.

[40] Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. Tight FPT Approximations for k-Median and k-Means. In *Proc. 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132, pages 42:1–42:14, 2019.

[41] Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. Tight FPT approximations for k-median and k-means. In *Proc. 46th International Colloquium on Automata, Languages, and Programming (ICALP'19)*, volume 132 of *LIPIcs*, pages 42:1–42:14, 2019.

[42] Vincent Cohen-Addad and CS Karthik. Inapproximability of clustering in lp metrics. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 519–539. IEEE, 2019.

[43] Vincent Cohen-Addad, CS Karthik, and Euiwoong Lee. On approximability of clustering problems without candidate centers. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2635–2648. SIAM, 2021.

[44] Vincent Cohen-Addad, Philip N Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in euclidean and minor-free metrics. *SIAM Journal on Computing*, 48(2):644–667, 2019.

[45] Vincent Cohen-Addad, Kasper Green Larsen, David Saulpic, and Chris Schwiegelshohn. Towards optimal lower bounds for k-median and k-means coresets. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC'22)*, pages 1038–1051, 2022.

[46] Vincent Cohen-Addad and Euiwoong Lee. Johnson coverage hypothesis: Inapproximability of k-means and k-median in lp-metrics. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1493–1530. SIAM, 2022.

[47] Vincent Cohen-Addad, Marcin Pilipczuk, and Michal Pilipczuk. Efficient approximation schemes for uniform-cost clustering problems in planar graphs. In *27th Annual European Symposium on Algorithms (ESA 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[48] Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In Samir Khuller and Virginia Vassilevska Williams, editors, *Proc. 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC'21)*, pages 169–182, 2021.

[49] Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC'21)*, pages 169–182, 2021.

[50] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[51] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[52] Marek Cygan, Fedor V. Fomin, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[53] Sanjoy Dasgupta. The hardness of k-means clustering. Technical Report CS2008-0916, University of California, San Diego, San Diego, CA, 2008.

[54] Shichuan Deng and Qianfan Zhang. Ordered k-median with outliers. In Amit Chakrabarti and Chaitanya Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, (APPROX/RANDOM'22)*, volume 245 of *LIPIcs*, pages 34:1–34:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[55] Hu Ding and Jinhui Xu. A unified framework for clustering constrained data without locality property. *Algorithmica*, 82(4):808–852, 2020.

[56] Petros Drineas, Alan M. Frieze, Ravi Kannan, Santosh S. Vempala, and V. Vinay. Clustering large graphs via the singular value decomposition. *Mach. Learn.*, 56(1-3):9–33, 2004.

[57] David Eisenstat, Philip N Klein, and Claire Mathieu. Approximating k-center in planar graphs. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 617–627. SIAM, 2014.

[58] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.

[59] Tomás Feder and Daniel H. Greene. Optimal algorithms for approximate clustering. In *Proc. 20th Annual ACM Symposium on Theory of Computing (STOC'88)*, pages 434–444, 1988.

[60] Andreas Emil Feldmann and Dániel Marx. The parameterized hardness of the k-center problem in transportation networks. *Algorithmica*, 82(7):1989–2005, 2020.

[61] Arnold Filtser and Hung Le. Low treewidth embeddings of planar and minor-free metrics. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS'22)*, pages 1081–1092. IEEE, 2022.

[62] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer-Verlag, 2006.

[63] Fedor V. Fomin, Dieter Kratsch, and Gerhard J. Woeginger. Exact (exponential) algorithms for the dominating set problem. In Juraj Hromkovic, Manfred Nagl, and Bernhard Westfechtel, editors, *Graph-Theoretic Concepts in Computer Science, 30th International Workshop, WG 2004, Bad Honnef, Germany, June 21-23, 2004, Revised Papers*, volume 3353 of *Lecture Notes in Computer Science*, pages 245–256. Springer, 2004.

[64] Eli Fox-Epstein, Philip N Klein, and Aaron Schild. Embedding planar graphs into low-treewidth graphs with applications to efficient approximation schemes for metric problems. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1069–1088. SIAM, 2019.

[65] Zachary Friggstad, Mohsen Rezapour, and Mohammad R Salavatipour. Local search yields a ptas for k-means in doubling metrics. *SIAM Journal on Computing*, 48(2):452–480, 2019.

[66] Satoru Fujishige. *Submodular functions and optimization*. Elsevier, 2005.

[67] Ameet Gadekar and Tanmay Inamdar. Dimension-free parameterized approximation schemes for hybrid clustering. *arXiv preprint arXiv:2501.03663*, 2025.

[68] Naveen Garg, Anupam Gupta, Stefano Leonardi, and Piotr Sankowski. Stochastic analyses for online combinatorial optimization problems. In Shang-Hua Teng, editor, *19th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 942–951. SIAM, 2008.

[69] Mehrdad Ghadiri, Samira Samadi, and Santosh Vempala. Socially fair k-means clustering. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 438–448, 2021.

[70] Mehrdad Ghadiri, Mohit Singh, and Santosh S Vempala. Constant-factor approximation algorithms for socially fair k-clustering. *arXiv preprint arXiv:2206.11210*, 2022.

[71] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

[72] Kishen N. Gowda, Thomas W. Pensyl, Aravind Srinivasan, and Khoa Trinh. Improved bi-point rounding algorithms and a golden barrier for k-median. In Nikhil Bansal and Viswanath Nagarajan, editors, *34th ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 987–1011. SIAM, 2023.

[73] Dishant Goyal and Ragesh Jaiswal. Tight fpt approximation for socially fair clustering. *Information Processing Letters*, page 106383, 2023.

[74] Dishant Goyal and Ragesh Jaiswal. Tight fpt approximation for socially fair clustering. *Information Processing Letters*, 182:106383, 2023.

[75] Fabrizio Grandoni, Anupam Gupta, Stefano Leonardi, Pauli Miettinen, Piotr Sankowski, and Mohit Singh. Set covering with our eyes closed. *SIAM J. Comput.*, 42(3):808–830, 2013.

[76] Fabrizio Grandoni, Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Rakesh Venkat. A refined approximation for Euclidean k-means. *Inf. Process. Lett.*, 176:106251, 2022.

[77] Fabrizio Grandoni, Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Rakesh Venkat. A refined approximation for euclidean k-means. *Inf. Process. Lett.*, 176:106251, 2022.

[78] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

[79] Sudipto Guha and Samir Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31(1):228–248, 1999.

[80] Anupam Gupta, Robert Krauthgamer, and James R Lee. Bounded geometries, fractals, and low-distortion embeddings. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 534–543. IEEE, 2003.

[81] Anupam Gupta, Martin Pál, R. Ravi, and Amitabh Sinha. Sampling and cost-sharing: Approximation algorithms for stochastic optimization problems. *SIAM J. Comput.*, 40(5):1361–1401, 2011.

[82] Shalmoli Gupta. *Approximation algorithms for clustering and facility location problems*. PhD thesis, University of Illinois at Urbana-Champaign, USA, 2018.

[83] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proc. 36th annual ACM symposium on Theory of Computing (STOC'04)*, pages 291–300, 2004.

[84] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proc. 36th Annual ACM Symposium on Theory of Computing (STOC'04)*, page 291–300, 2004.

[85] Juha Heinonen. *Lectures on analysis on metric spaces*. Springer Science & Business Media, 2001.

[86] Lingxiao Huang and Nisheeth K Vishnoi. Coresets for clustering in euclidean spaces: importance sampling is nearly optimal. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC'20)*, pages 1416–1429, 2020.

[87] Nicole Immorlica, David R. Karger, Maria Minkoff, and Vahab S. Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In J. Ian Munro, editor, *15th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 691–700. SIAM, 2004.

[88] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM (JACM)*, 48(4):761–777, 2001.

[89] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *Journal of the ACM (JACM)*, 50(6):795–824, 2003.

[90] 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 (JACM)*, 48(2):274–296, 2001.

[91] Ioannis Katsikarelis, Michael Lampis, and Vangelis Th Paschos. Structural parameters, tight bounds, and approximation for (k, r)-center. *Discrete Applied Mathematics*, 264:90–117, 2019.

[92] Stavros G Kolliopoulos and Satish Rao. A nearly linear-time approximation scheme for the euclidean k-median problem. *SIAM Journal on Computing*, 37(3):757–782, 2007.

[93] Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC'18)*, pages 646–659. ACM, 2018.

[94] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear-time approximation schemes for clustering problems in any dimensions. *Journal of the ACM (JACM)*, 57(2):1–32, 2010.

[95] Piyush Kumar and E. Alper Yildirim. An algorithm and a core set result for the weighted euclidean one-center problem. *INFORMS J. Comput.*, 21(4):614–629, 2009.

[96] G. Laporte, S. Nickel, and F. S. da Gama. *Location Science*. Springer, 2015.

[97] Yin Tat Lee, Aaron Sidford, and Santosh S Vempala. Efficient convex optimization with membership oracles. In *Conference On Learning Theory (COLT'18)*, pages 1292–1294. PMLR, 2018.

[98] Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Inf. Comput.*, 222:45–58, 2013.

[99] Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. *SIAM J. Comput.*, 45(2):530–547, 2016.

[100] Inge Li Gørtz and Anthony Wirth. Asymmetry in k-center variants. In *Approximation, Randomization, and Combinatorial Optimization.. Algorithms and Techniques*, pages 59–70. Springer, 2003.

[101] László Lovász and Santosh Vempala. Fast algorithms for logconcave functions: Sampling, rounding, integration and optimization. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 57–68. IEEE, 2006.

[102] Mohammad Mahdian, Yinyu Ye, and Jiawei Zhang. Approximation algorithms for metric facility location problems. *SIAM J. Comput.*, 36(2):411–432, 2006.

[103] Yury Makarychev and Ali Vakilian. Approximation algorithms for socially fair clustering. In Mikhail Belkin and Samory Kpotufe, editors, *Proceedings of Thirty Fourth Conference on Learning Theory*,

volume 134 of *Proceedings of Machine Learning Research*, pages 3246–3264. PMLR, 15–19 Aug 2021.

[104] Jirı Matoušek. On approximate geometric k-clustering. *Discrete & Computational Geometry*, 24(1):61–84, 2000.

[105] Adam Meyerson. Online facility location. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 426–431. IEEE Computer Society, 2001.

[106] Viswanath Nagarajan, Baruch Schieber, and Hadas Shachnai. The Euclidean *k*-supplier problem. *Math. Oper. Res.*, 45(1):1–14, 2020.

[107] S. Nickel and J. Puerto. *Location Theory*. Springer Science & Business Media, 2005.

[108] Rafail Ostrovsky, Yuval Rabani, Leonard J Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. *Journal of the ACM (JACM)*, 59(6):1–22, 2013.

[109] Ján Plesník. A heuristic for the p-center problems in graphs. *Discrete Applied Mathematics*, 17(3):263–268, 1987.

[110] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[111] David B. Shmoys, Chaitanya Swamy, and Retsef Levi. Facility location with service installation costs. In J. Ian Munro, editor, *15th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 1088–1097. SIAM, 2004.

[112] David B. Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems (extended abstract). In Frank Thomson Leighton and Peter W. Shor, editors, *29th Annual ACM Symposium on the Theory of Computing, STOC 1997, El Paso, Texas, USA, May 4-6, 1997*, pages 265–274. ACM, 1997.

[113] Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. In *Proc. 59th IEEE Annual Symposium on Foundations of Computer Science (FOCS'18)*, pages 802–813, 2018.

[114] Zoya Svitkina and Éva Tardos. Facility location with hierarchical facility costs. In *SODA*, volume 6, pages 153–161, 2006.

[115] Zoya Svitkina and Éva Tardos. Facility location with hierarchical facility costs. *ACM Trans. Algorithms*, 6(2):37:1–37:22, 2010.

[116] Amnon Ta-Shma. Explicit, almost optimal, ϵ-balanced codes. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proc. 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC'17)*, pages 238–251. ACM, 2017.

[117] Arie Tamir. The k-centrum multi-facility location problem. *Discrete Applied Mathematics*, 109(3):293–307, 2001.

[118] Di Wu, Jinhui Xu, and Jianxin Wang. A ptas framework for clustering problems in doubling metrics. In *International Computing and Combinatorics Conference*, pages 384–397. Springer, 2023.

[119] Jiawei Zhang, Bo Chen, and Yinyu Ye. A multiexchange local search algorithm for the capacitated facility location problem. *Mathematics of Operations Research*, 30(2):389–403, 2005.