Coresets for Clustering in Excluded-minor Graphs and Beyond

Vladimir Braverman* Shaofeng H.-C. Jiang† Robert Krauthgamer† Xuan Wu*

Abstract

Coresets are modern data-reduction tools that are widely used in data analysis to improve efficiency in terms of running time, space and communication complexity. Our main result is a fast algorithm to construct a small coreset for k-Median in (the shortest-path metric of) an excluded-minor graph. Specifically, we give the first coreset of size that depends only on k, ϵ and the excluded-minor size, and our running time is quasi-linear (in the size of the input graph).

The main innovation in our new algorithm is that is iterative; it first reduces the n input points to roughly $O(\log n)$ reweighted points, then to $O(\log\log n)$, and so forth until the size is independent of n. Each step in this iterative size reduction is based on the importance sampling framework of Feldman and Langberg (STOC 2011), with a crucial adaptation that reduces the number of *distinct points*, by employing a terminal embedding (where low distortion is guaranteed only for the distance from every terminal to all other points). Our terminal embedding is technically involved and relies on shortest-path separators, a standard tool in planar and excluded-minor graphs.

Furthermore, our new algorithm is applicable also in Euclidean metrics, by simply using a recent terminal embedding result of Narayanan and Nelson, (STOC 2019), which extends the Johnson-Lindenstrauss Lemma. We thus obtain an efficient coreset construction in high-dimensional Euclidean spaces, thereby matching and simplifying state-of-the-art results (Sohler and Woodruff, FOCS 2018; Huang and Vishnoi, STOC 2020).

In addition, we also employ terminal embedding with additive distortion to obtain small coresets in graphs with bounded highway dimension, and use applications of our coresets to obtain improved approximation schemes, e.g., an improved PTAS for planar k-Median via a new centroid set.

*Johns Hopkins University. Email: {vova@cs.jhu.edu, xwu71@jh.edu}

†Weizmann Institute of Science. Work partially supported by ONR Award N00014-18-1-2364, the Israel Science Foundation grant #1086/18, and a Minerva Foundation grant. Part of this work was done while some of the authors were visiting the Simons Institute for the Theory of Computing. Email: {shaofeng.jiang, robert.krauthgamer}@weizmann.ac.il

1 Introduction

Coresets are modern tools for efficient data analysis that have become widely used in theoretical computer science, machine learning, networking and other areas. This paper investigates coresets for the metric k-Median problem that can be defined as follows. Given an *ambient* metric space $M = (V, d)$ and a *weighted* set $X \subseteq V$ with weight function $w : X \to \mathbb{R}_+$, the goal is to find a set of k *centers* $C \subseteq V$ that minimizes the total cost of connecting every point to a center in C:

$$\text{cost}(X, C) := \sum_{x \in X} w(x) \cdot d(x, C),$$

where $d(x, C) := \min_{y \in C} d(x, y)$ is the distance to the closest center. An ϵ-*coreset* for k-Median on X is a weighted subset $D \subseteq X$, such that

$$\forall C \subseteq V, |C| = k, \qquad \text{cost}(D, C) \in (1 \pm \epsilon) \cdot \text{cost}(X, C).$$

We note that many papers study a more general problem, (k, z)-Clustering, where inside the cost function each distance is raised to power z. We focus on k-Median for sake of exposition, but most of our results easily extend to (k, z)-Clustering.

Small coresets are attractive since one can solve the problem on D instead of X and, as a result, improve time, space or communication complexity of downstream applications [LBK13, LFKF17, FSS20]. Thus, one of the most important performance measures of a coreset D is its *size*, i.e., the number of distinct points in it, denoted $\|D\|_0$.[1] Har-Peled and Mazumdar [HM04] introduced the above definition and designed the first coreset for k-Median in Euclidean spaces ($V = \mathbb{R}^m$ with ℓ_2 norm), and since their work, designing small coresets has become a flourishing research direction, including not only k-Median and (k, z)-Clustering e.g. [HK07, Che09, LS10, FL11, SW18, HV20, FSS20], but also many other important problems, such as subspace approximation/PCA [FFS06, FMSW10, FSS20], projective clustering [FL11, VX12, FSS20], regression [MJF19], density estimation [KL19, PT19], ordered weighted clustering [BJKW19], and fair clustering [SSS19, HJV19].

Many modern coreset constructions stem from a fundamental framework proposed by Feldman and Langberg [FL11], extending the importance sampling approach of Langberg and Schulman [LS10]. In this framework [FL11], the size of an ϵ-coreset for k-Median is bounded by $O(\text{poly}(k/\epsilon) \cdot \text{sdim})$, where sdim is the shattering (or VC) dimension of the family of distance functions. For a general metric space (V, d), a direct application of [FL11] results in a coreset of size $O_{k,\epsilon}(\log |V|)$, which is tight in the sense that in some instances, every coreset must have size $\Omega(\log |V|)$ [BBH+20].Therefore, to obtain coresets of size independent of the data set X, we have to restrict our attention to specific metric spaces, which raises the following fundamental question.

Question 1.1. *Identify conditions on a data set X from metric space (V, d) that guarantee the existence (and efficient construction) of an ϵ-coreset for k-Median of size $O_{\epsilon,k}(1)$?*

This question has seen major advances recently. Coresets of size independent of X (and V) were obtained, including efficient algorithms, for several important special cases: high-dimensional Euclidean spaces [SW18, FKW19, HV20] (i.e., independently of the Euclidean dimension), metrics with bounded doubling dimension [HJLW18], and shortest-path metric of bounded-treewidth graphs [BBH+20].

[1] For a weighted set X, we denote by $\|X\|_0$ the number of *distinct* elements, by $\|X\|_1$ its total weight.

1.1 Our Results

Overview We make significant progress on this front (Question 1.1) by designing new coresets for k-MEDIAN in three very different types of metric spaces. Specifically, we give (i) the first $O_{\epsilon,k}(1)$-size coreset for excluded-minor graphs; (ii) the first $O_{\epsilon,k}(1)$-size coreset for graphs with bounded highway dimension; and (iii) a simplified state-of-the-art coreset for high-dimensional Euclidean spaces (i.e., coreset-size independent of the Euclidean dimension with guarantees comparable to [HV20] but simpler analysis.)

Our coreset constructions are all based on the well-known importance sampling framework of [FL11], but with subtle deviations that introduce significant advantages. Our first technical idea is to relax the goal of computing the final coreset in one shot: we present a general reduction that turns an algorithm that computes a coreset of size $O(\text{poly}(k/\epsilon)\log\|X\|_0)$ into an algorithm that computes a coreset of size $O(\text{poly}(k/\epsilon))$. The reduction is very simple and efficient, by straightforward iterations. Thus, it suffices to construct a coreset of size $O(\text{poly}(k/\epsilon)\log\|X\|_0)$. We construct this using the importance sampling framework [FL11], but applied in a subtly different way, called terminal embedding, in which distances are slightly distorted, trading accuracy for (hopefully) a small shattering dimension. It still remains to bound the shattering dimension, but we are now much better equipped — we can distort the distances (design a new embedding or employ a known one), and we are content with dimension bound $O_{k,\epsilon}(\log\|X\|_0)$, instead of the usual $O_{k,\epsilon}(1)$.

We proceed to present each of our results and its context-specific background, see also Table 1 for summary, and then describe our techniques at a high-level in Section 1.2.

Table 1: our results of ϵ-coresets for k-MEDIAN in various types of metric spaces $M(V,d)$ with comparison to previous works. By graph metric, we mean the shortest-path metric of an edge-weighted graph $G=(V,E)$. Corollary 4.18 (and [HV20]) also work for general (k,z)-CLUSTERING, but we list the result for k-MEDIAN ($z=1$) only.

Metric space		Coreset size[2]	Reference
General metrics		$\tilde{O}(\epsilon^{-2}k\log\lvert V\rvert)$	[FL11]
Graph metrics	Bounded treewidth	$\tilde{O}(\epsilon^{-2}k^3)$	[BBH+20]
	Excluding a fixed minor	$\tilde{O}(\epsilon^{-4}k^2)$	Corollary 4.2
	Bounded highway dimension	$\tilde{O}(k^{O(\log(1/\epsilon))})$	Corollary 4.25
Euclidean $\mathbb{R}^m$	Dimension-dependent	$\tilde{O}(\epsilon^{-2}km)$	[FL11]
	Dimension-free	$\tilde{O}(\epsilon^{-4}k)$	[HV20], Corollary 4.18

Coresets for Clustering in Graph Metrics k-MEDIAN clustering in graph metrics, i.e. shortest-path metric of graphs, is a central task in data mining of spatial networks (e.g., planar networks such as road networks) [SL97, YM04], and has applications in various location optimization problems, such as placing servers on the Internet [LGI+99, JJJ+00] (see also a survey [TFL83]), and in data analysis methods [RMJ07, CZQ+08]. We obtain new coresets for excluded-minor graphs and new coresets for graphs of bounded highway dimension. The former generalize planar graphs and the latter capture the structure of transportation networks.

[2]Throughout, the notation $\tilde{O}(f)$ hides $\text{poly}\log f$ factors, and $O_m(f)$ hides factors that depend on m.

Coresets for Excluded-minor Graphs A *minor* of graph G is a graph H obtained from G by a sequence of edge deletions, vertex deletions or edge contractions. We are interested in graphs G that exclude a fixed graph H as a minor, i.e., they do not contain H as a minor. Excluded-minor graphs have found numerous applications in theoretical computer science and beyond and they include, for example, planar graphs and bounded-treewidth graphs. Besides its practical importance, k-Median in planar graphs received significant attention in approximation algorithms research [Tho05, CKM19, CPP19]. Our framework yields the first ϵ-coreset of size $O_{k,\epsilon}(1)$ for k-Median in excluded-minor graphs, see Corollary 4.2 for details. Such a bound was previously known only for the special case of bounded-treewidth graphs [BBH+20]. We stress that our technical approach is significantly different from [BBH+20]; we introduce a novel iterative construction and a relaxed terminal embedding of excluded-minor graph metrics (see Section 1.2), and overall bypass bounding the shattering dimension by $O(1)$ (which is the technical core in [BBH+20]).

Coresets for Graphs with Bounded Highway Dimension Due to the tight relation to road networks, graphs of bounded highway dimension is another important family for the study of clustering in graph metrics. The notion of highway dimension was first proposed by [AFGW10] to measure the complexity of transportation networks such as road networks and airline networks. Intuitively, it captures the fact that going from any two far-away cities A and B, the shortest path between A and B always goes through a small number of connecting hub cities. The formal definition of highway dimension is given in Definition 4.20, and we compare related versions of definitions in Remark 4.21. The study of highway dimension was originally to understand the efficiency of heuristics for shortest path computations [AFGW10], while subsequent works also study approximation algorithms for optimization problems such as TSP, Steiner Tree [FFKP18] and k-Median [BKS18]. We show the first coreset for graphs with bounded highway dimension, and as we will discuss later it can be applied to design new approximation algorithms. The formal statement can be found in Corollary 4.25.

Coresets for High-dimensional Euclidean Space The study of coresets for k-Median (and more generally (k,z)-Clustering) in Euclidean space $\mathbb{R}^m$ spans a rich line of research. The first coreset for k-Median in Euclidean spaces, given by [HM04], has size $O(k\epsilon^{-m}\log n)$ where $n = \|X\|_1$, and the $\log n$ factor was shaved by a subsequent work [HK07]. The exponential dependence on the Euclidean dimension m was later improved to $\mathrm{poly}(km/\epsilon)$ [LS10], and to $O(km/\epsilon^2)$ [FL11]. Very recently, the first coreset for k-Median of size $\mathrm{poly}(k/\epsilon)$, which is *independent* of the Euclidean dimension m,[3] was obtained by [SW18] (see also [FKW19]).[4] This was recently improved in [HV20], which designs a (much faster) near-linear time construction for (k,z)-Clustering, with slight improvements in the coreset size and the (often useful) additional property that the coreset is a subset of X. Our result extends this line of research; an easy application of our new framework yields a near-linear time construction of coreset of size $\mathrm{poly}(k/\epsilon)$, which too is independent of the dimension m. Compared to the state of the art [HV20], our result achieves essentially the same size bound, while greatly simplifying the analysis. A formal statement and detailed comparison with [HV20] can be found in Corollary 4.18 and Remark 4.19.

Applications: Improved Approximation Schemes We apply our coresets to design approximation schemes for k-Median in shortest-path metrics of planar graphs and graphs with bounded

[3] Dimension-independent coresets were obtained earlier for Euclidean k-Means [BFL16, FSS20], however these do not apply to k-Median.

[4] The focus of [SW18] is on k-Median, but the results extend to (k,z)-Clustering.

highway dimension. In particular, we give an FPT-PTAS, parameterized by k and ϵ, in graphs with bounded highway dimension (Corollary 5.2), and a PTAS in planar graphs (Corollary 5.9). Both algorithms run in time near-linear in $|V|$, and improve previous results in the corresponding settings.

The PTAS for k-MEDIAN in planar graphs is obtained using a new centroid-set result. A *centroid set* is a subset of V that contains centers giving a $(1+\epsilon)$-approximate solution. We obtain centroid sets of size *independent* of the input X in planar graphs, which improves a recent size bound $(\log |V|)^{O(1/\epsilon)}$ [CPP19], and moreover runs in time near-linear in $|V|$. This centroid set can be found in Theorem 5.4.

1.2 Technical Contributions

Iterative Size Reduction This technique is based on an idea so simple that it may seem too naive: Basic coreset constructions have size $O_{k,\epsilon}(\log n)$, so why not apply it repeatedly, to obtain a coreset of size $O_{k,\epsilon}(\log\log n)$, then $O_{k,\epsilon}(\log\log\log n)$ and so on? One specific example is the size bound $O(\epsilon^{-2}k\log n)$ for a general n-point metric space [FL11], where this does not work because $n = |V|$ is actually the size of the *ambient* space, irrespective of the *data* set X. Another example is the size bound $O(\epsilon^{-m}k\log n)$ for Euclidean space $\mathbb{R}^m$ [HM04], where this does not work because $n = \|X\|_1$ is the total weight of the data points X, which coresets do not reduce (to the contrast, they maintain it). These examples suggest that one should avoid two pitfalls: dependence on V and dependence on the total weight.

We indeed make this approach work by requiring an algorithm $\mathcal{A}$ that constructs a coreset of size $O(\log \|X\|_0)$, which is *data-dependent* (recall that $\|X\|_0$ is the number of *distinct* elements in a weighted set X). Specifically, we show in Theorem 3.1 that, given an algorithm $\mathcal{A}$ that constructs an ϵ'-coreset of size $O(\mathrm{poly}(k/\epsilon')\log\|X\|_0)$ for every ϵ' and $X \subseteq V$, one can obtain an ϵ-coreset of size $\mathrm{poly}(k/\epsilon)$ by simply applying $\mathcal{A}$ iteratively. It follows by setting ϵ' carefully, so that it increases quickly and eventually $\epsilon' = O(\epsilon)$. See Section 3.1 for details.

Not surprisingly, the general idea of applying the sketching/coreset algorithm iteratively was also used in other related contexts (e.g. [LMP13, CW15, MSSW18]). Moreover, a related two-step iterative construction was applied in a recent coreset result [HV20]. Nevertheless, the exact implementation of iterative size reduction in coresets is unique in the literature. As can be seen from our results, this reduction fundamentally helps to achieve new or simplified coresets of size *independent* of data set. We expect the iterative size reduction to be of independent interest to future research.

Terminal Embeddings To employ the iterative size reduction, we need to construct coresets of size $\mathrm{poly}(k/\epsilon)\cdot\log\|X\|_0$. Unfortunately, a direct application of [FL11] yields a bound that depends on the number of vertices $|V|$, irrespective of X. To bypass this limitation, the framework of [FL11] is augmented (in fact, we use a refined framework proposed in [FSS20]), to support controlled modifications to the distances $d(\cdot,\cdot)$. As explained more formally in Section 3.2, one represents these modifications using a set of functions $\mathcal{F} = \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$, that corresponds to the modified distances from each x, i.e., $f_x(\cdot) \leftrightarrow d(x,\cdot)$. Many previous papers [LS10, FL11, BFL16, FSS20] work directly with the distances and use the function set $\mathcal{F} = \{f_x(\cdot) = d(x,\cdot) \mid x \in X\}$, or a more sophisticated but still direct variant of hyperbolic balls (where each f_x is an affine transformation of $d(x,\cdot)$). A key difference is that we use a "proxy" function set $\mathcal{F}$, where each $f_x(\cdot) \approx d(x,\cdot)$. This introduces a tradeoff between the approximation error (called distortion) and the shattering dimension of $\mathcal{F}$ (which controls the number of samples), and overall results in a smaller coreset. Such tradeoff was first used in [HJLW18] to obtain small coresets for doubling spaces, and was

recently used in [HV20] to reduce the coreset size for Euclidean spaces. This proxy function set may be alternatively viewed as a *terminal embedding* on X, in which both the distortion of distances (between X and all of V) and the shattering dimension are controlled.

We then consider two types of terminal embeddings $\mathcal{F}$. The first type (Section 3.3) maintains $(1+\epsilon)$-multiplicative distortion of the distances. When this embedding achieves dimension bound $O(\text{poly}(k/\epsilon)\log\|X\|_0)$, we combine it with the aforementioned iterative size reduction, to further reduce the size to be independent of X. It remains to actually design embeddings of this type, which we achieve (as explained further below), for excluded-minor graphs and for Euclidean spaces, and thus we overall obtain $O_{\epsilon,k}(1)$-size coresets in both settings. Our second type of terminal embeddings $\mathcal{F}$ (Section 3.4) maintains additive distortion on top of the multiplicative one. We design embeddings of this type (as explained further below) for graphs with bounded highway dimension; these embeddings have shattering dimension $\text{poly}(k/\epsilon)$, and thus we overall obtain $O_{\epsilon,k}(1)$-size coresets even without the iterative size reduction. We report our new terminal embeddings in Table 2.

Table 2: New terminal embeddings $\mathcal{F}$ for different metrics spaces. The reported distortion bound is the upper bound on $f_x(c)$, in addition to the lower bound $f_x(c) \geq d(x,c)$. The embeddings of graphs with bounded highway dimension, called here "highway graphs" for short, are defined with respect to a given $S \subseteq V$ (see Lemma 4.22).

Metric space	Dimension $\text{sdim}_{\max}(\mathcal{F})$	Distortion	Result
Euclidean	$O(\epsilon^{-2}\log\|X\|_0)$	$(1+\epsilon)\cdot d(x,c)$	Lemma 4.16
Excluded-minor graphs	$\tilde{O}(\epsilon^{-2}\log\|X\|_0)$	$(1+\epsilon)\cdot d(x,c)$	Lemma 4.1
Highway graphs	$O(\lvert S\rvert^{O(\log(1/\epsilon))})$	$(1+\epsilon)\cdot d(x,c)+\epsilon\cdot d(x,S)$	Lemma 4.22

Terminal Embedding for Euclidean Spaces Our terminal embedding for Euclidean spaces is surprisingly simple, and is a great showcase for our new framework. In a classical result [FL11], it has been shown that $\text{sdim}_{\max}(\mathcal{F}) = O(m)$ for Euclidean distance in $\mathbb{R}^m$ without distortion. On the other hand, we notice a terminal embedding version of Johnson-Lindenstrauss Lemma was discovered recently [NN19]. Our terminal embedding bound (Lemma 4.16) follows by directly combining these two results, see Section 4.3 for details.

We note that without our iterative size reduction technique, plugging in the recent terminal Johnson-Lindenstrauss Lemma [NN19] into classical importance sampling frameworks, such as [FL11, FSS20] does not yield any interesting coreset. Furthermore, the new terminal Johnson-Lindenstrauss Lemma was recently used in [HV20] to design coresets for high-dimensional Euclidean spaces. Their size bounds are essentially the same as ours, however they go through a complicated analysis to directly show a shattering dimension bound $\text{poly}(k/\epsilon)$. This complication is not necessary in our method, because by our iterative size reduction it suffices to show a very loose $O_{k,\epsilon}(\log\|X\|_0)$ dimension bound, and this follows immediately from the Johnson-Lindenstrauss result.

Terminal Embedding for Excluded-minor Graphs The technical core of the terminal embedding for excluded-minor graphs is how to bound the shattering dimension. In our proof, we reduce the problem of bounding the shattering dimension into finding a representation of the distance functions on $X \times V$ as a set of *min-linear* functions. Specifically, we need to find for each x

a min-linear function $g_x : \mathbb{R}^s \to \mathbb{R}$ of the form $g_x(t) = \min_{1 \le i \le s}\{a_i t_i + b_i\}$, where $s = O(\log \|X\|_0)$, such that $\forall c \in V$, there is $t \in \mathbb{R}^s$ with $d(x, c) = g_x(t)$.

The central challenge is how to relate the graph structure to the structure of shortest paths $d(x, c)$. To demonstrate how we relate them, we start with discussing the simple special case of bounded treewidth graphs. For bounded treewidth graphs, the vertex separator theorem is applied to find a subset $P \subseteq V$, through which the shortest path $x \rightsquigarrow y$ has to pass. This translates into the following

$$d(x, c) = \min_{p \in P}\{d(x, p) + d(p, c)\},$$

and for each $x \in X$, we can use this to define the desired min-linear function $g_x(d(p_1, c), \ldots, d(p_m, c))$ $= d(x, c)$, where we write $P = \{p_1, \ldots, p_m\}$.

However, excluded-minor graphs do not have small vertex separator, and we use the shortest-path separator [Tho04, AG06] instead. Now assume for simplicity that the shortest paths $x \rightsquigarrow c$ all pass through a fixed shortest path l. Because l itself is a shortest path, we know

$$\forall x \in X, c \in V, \quad d(x, c) = \min_{u_1, u_2 \in l}\{d(x, u_1) + d(u_1, u_2) + d(u_2, c)\}.$$

Since l can have many (i.e. $\omega(\log \|X\|_0)$) points, we need to discretize l by designating $\mathrm{poly}(\epsilon^{-1})$ *portals* P_x^l on l for each $x \in X$ (and similarly P_c^l for $c \in V$). This only introduces $(1 + \epsilon)$ distortion to the distance, which we can afford.

Then we create $d'_x : l \to \mathbb{R}_+$ to approximate $d(x, u)$'s, using distances from x to the portals P_x^l (and similarly for $d(c, u)$). Specifically, for the sake of presentation, assume $P_x^l = \{p_1, p_2, p_3\}$ $(p_1 \le p_2 \le p_3)$, interpret l as interval $[0, 1)$, then for $u \in [0, p_1)$, define $d'_x(u) = d(x, 0)$, for $u \in [p_1, p_2)$, define $d'_x(u) = d(x, p_1)$, and so forth. Hence, each $d'_x(\cdot)$ is a piece-wise linear function of $O(|P_x^l|)$ pieces (again, similarly for $d'_c(\cdot)$), and this enables us to write

$$d(x, c) \approx d'(x, c) := \min_{u_1, u_2 \in P_x^l \cup P_c^l}\{d'_x(u_1) + d(u_1, u_2) + d'_c(u_2)\}.$$

Therefore, it suffices to find a min-linear representation for $d'(x, \cdot)$ for $x \in X$. However, the piece-wise linear structure of d'_x creates extra difficulty to define min-linear representations. To see this, still assume $P_x^l = \{p_1, p_2, p_3\}$. Then to determine $d'_x(u)$ for $u \in P_x^l \cup P_c^l$, we not only need to know $d(x, p_i)$ for $p_i \in P_x^l$, but also need to know which sub-interval $[p_i, p_{i+1})$ that u belongs to. (That is, if $u \in [p_1, p_2)$, then $d'_x(u) = d(x, p_1)$.) Hence, in addition to using distances $\{c\} \times P_c^l$ as variables of g_x, the relative ordering between points in $P_x^l \cup P_c^l$ is also necessary to evaluate $d'(x, c)$.

Because $c \in V$ can be arbitrary, we cannot simply "remember" the ordering in g_x. Hence, we "guess" this ordering, and for each fixed ordering we can write g_x as a min-linear function of few variables. Luckily, we can afford the "guess" since $|P_x^l \cup P_c^l| = \mathrm{poly}(\epsilon^{-1})$ which is independent of X. A more detailed overview can be found in Section 4.1.

Terminal Embedding for Graphs with Bounded Highway Dimension In addition to a $(1 + \epsilon)$ multiplicative error, the embedding for graphs with bounded highway dimension also introduces an additive error. In particular, for a given $S \subseteq V$, it guarantees that

$$\forall x \in X, c \in V, \quad d(x, c) \le f_x(c) \le (1 + \epsilon) \cdot d(x, c) + \epsilon \cdot d(x, S).$$

This terminal embedding is a direct consequence of a similar embedding from graphs with bounded highway dimension to graphs with bounded treewidth [BKS18], and a previous result about the

shattering dimension for graphs with bounded treewidth [BBH+20]. In our applications, we will choose S to be a constant approximate solution[5] $C^\star$ to k-MEDIAN. So the additive error becomes $\epsilon \cdot d(x, C^\star)$. In general, this term can still be much larger than $d(x, c)$, but the *collectively* error in the clustering objective is bounded. This observation helps us to obtain a coreset, and due to the additional additive error, the shattering dimension is already independent of X and hence no iterative size reduction is necessary.

1.3 Related Work

Approximation algorithms for metric k-MEDIAN have been extensively studied. In general metric spaces, it is NP-hard to approximate k-MEDIAN within a $1 + \frac{2}{e}$ factor [JMS02], and the state of the art is a $(2.675 + \epsilon)$-approximation [BPR+14]. In Euclidean space $\mathbb{R}^m$, k-MEDIAN is APX-hard if both k and the dimension m are part of the input [GI03]. However, PTAS's do exist if either k or dimension m is fixed [HM04, ARR98, CKM19, FRS19].

Tightly related to coresets, dimensionality reduction has also been studied for clustering in Euclidean spaces. Compared with coresets which reduce the data set size while keeping the dimension, dimensionality reduction aims to find a low-dimensional representation of data points (but not necessarily reduce the number of data points). As a staring point, a trivial application of Johnson-Lindenstrauss Lemma [JL84] yields a dimension bound $O(\epsilon^{-2} \log n)$ for (k, z)-CLUSTERING. For k-MEANS with $1+\epsilon$ approximation ratio, [CEM+15] showed an $O(k/\epsilon^2)$ dimension bound for data-oblivious dimension reduction and an $O(k/\epsilon)$ bound for the data-dependent setting. Moreover, the same work [CEM+15] also obtained a data-oblivious $O(\epsilon^{-2} \log k)$ dimension bound for k-MEANS with approximation ratio $9+\epsilon$. Very recently, [BBCA+19] obtained an $\tilde{O}(\epsilon^{-6}(\log k + \log\log n))$ dimension bound for k-MEANS and [MMR19] obtained an $O(\epsilon^{-2} \log \frac{k}{\epsilon})$ bound for (k, z)-CLUSTERING. Both of them used data-oblivious methods and have approximation ratio $1 + \epsilon$. Dimensionality reduction techniques are also used for constructing dimension-free coresets in Euclidean spaces [SW18, BBCA+19, HV20, FSS20].

2 Preliminaries

Notations Let $V^k := \{C \subseteq V : |C| \le k\}$ denote the collection of all subsets of V of size at most k. [6] For integer $n, i > 0$, let $\log^{(i)} n$ denote the i-th iterated logarithm of n, i.e. $\log^{(1)} n := \log n$ and $\log^{(i)} n := \log(\log^{(i-1)} n)$ $(i \ge 2)$. Define $\log^\star n$ as the number of times the logarithm is iteratively applied before the result is at most 1, i.e. $\log^\star n := 0$ if $n \le 1$ and $\log^\star n = 1 + \log^\star(\log n)$ if $n > 1$. For a weighted set S, denote the weight function as $w_S : S \to \mathbb{R}_+$. Let $\mathrm{OPT}_z(X)$ be the optimal objective value for (k, z)-CLUSTERING on X, and we call a subset $C \subseteq V$ an (α, β)-approximate solution for (k, z)-CLUSTERING on X if $|C| = \alpha k$ and $\mathrm{cost}_z(X, C) := \sum_{x \in X} w_X(x) \cdot (d(x, C))^z \le \beta \cdot \mathrm{OPT}_z(X)$.

Functional Representation of Distances We consider sets of functions $\mathcal{F}$ from V to $\mathbb{R}_+$. Specifically, we consider function sets $\mathcal{F} = \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ that is indexed by the weighted data set $X \subseteq V$, and intuitively $f_x(\cdot)$ is used to measure the distance from $x \in X$ to a point in V. Because we interpret f_x's as distances, for a subset $C \subseteq V$, we define $f_x(C) := \min_{c \in C} f_x(C)$, and

[5] in fact, a bi-criteria approximation suffices.

[6] Strictly speaking, V^k is the collection of all ordered k-tuples of V, but here we use it to denote the subsets. Note that tuples may contain repeated elements so the subsets in V^k are of size at most k.

define the clustering objective accordingly as

$$\mathrm{cost}_z(\mathcal{F}, C) := \sum_{f_x \in \mathcal{F}} w_{\mathcal{F}}(f_x) \cdot (f_x(C))^z.$$

In fact, in our applications, we will use $f_x(y)$ as a "close" approximation to d. We note that this functional representation is natural for k-Clustering, since the objective function only uses distances from X to every k-subset of V only. Furthermore, we do not require the triangle inequality to hold for such functional representations.

Shattering Dimension For $c \in V, r \geq 0$, define $B_{\mathcal{F}}(c, r) := \{f \in \mathcal{F} : f(c) \leq r\}$. We emphasize that c is from the ambient space V in addition to the data set X. Intuitively, $B_{\mathcal{F}}(c, r)$ is the ball centered at c with radius r when the f functions are used to measure distances. For example, consider $X = V$ and let $f_x(\cdot) := d(x, \cdot)$ for $x \in V$. Then $B_{\mathcal{F}}(c, r) = \{f_x \in \mathcal{F} : d(c, x) \leq r\}$, which corresponds to the metric ball centered at c with radius r.

We introduce the notion of shattering dimension in Definition 2.1. In fact, the shattering dimension may be defined with respect to any set system [Har11], but we do not need this generality here and thus we consider only the shattering dimension of the "metric balls" system.

Definition 2.1 (Shattering Dimension [Har11])**.** Suppose $\mathcal{F}$ is a set of functions from V to $\mathbb{R}_+$. The shattering dimension of $\mathcal{F}$, denoted as $\mathrm{sdim}(\mathcal{F})$, is the smallest integer t, such that for every $\mathcal{H} \subseteq \mathcal{F}$ with $|\mathcal{H}| \geq 2$,

$$\forall \mathcal{H} \subseteq \mathcal{F}, |\mathcal{H}| \geq 2, \quad |\{B_{\mathcal{H}}(c, r) : c \in V, r \geq 0\}| \leq |\mathcal{H}|^t. \tag{1}$$

The shattering dimension is tightly related to the well-known VC-dimension [VC71], and they are equal to each other up to a logarithmic factor [Har11, Corollary 5.12, Lemma 5.14]. In our application, we usually do not use $\mathrm{sdim}(\mathcal{F})$ directly. Instead, given a point weight $v : X \to \mathbb{R}_+$, we define $\mathcal{F}_v := \{f_x \cdot v(x) \mid x \in X\}$, and then consider the maximum of $\mathrm{sdim}(\mathcal{F}_v)$ over all possible v, defined as $\mathrm{sdim}_{\max}(\mathcal{F}) := \max_{v:X\to\mathbb{R}_+} \mathrm{sdim}(\mathcal{F}_v)$.

3 Framework

We present our general framework for constructing coresets. Our first new idea is a generic reduction, called iterative size reduction, through which it suffices to find a coreset of size $O(\log \|X\|_0)$ only in order to get a coreset of size independent of X. This general reduction greatly simplifies the coreset construction, and in particular, as we will see, "old" techniques such as importance sampling gains new power and becomes useful for new settings such as excluded-minor graphs.

Roughly speaking, the iterative size reduction turns a coreset construction algorithm $\mathcal{A}(X, \epsilon)$ with size $O(\mathrm{poly}(\epsilon^{-1}k) \cdot \log \|X\|_0)$ into a construction $\mathcal{A}'(X, \epsilon)$ with size $\mathrm{poly}(\epsilon^{-1}k)$. To define $\mathcal{A}'$, we simply iteratively apply $\mathcal{A}$, i.e. $X_i := \mathcal{A}(X_{i-1}, \epsilon_i)$, and terminate when $\|X_i\|_0$ does not decrease. However, if $\mathcal{A}$ is applied for t times in total, the error of the resulted coreset is accumulated as $\sum_{i=1}^{t} \epsilon_t$. Hence, to make the error bounded, we make sure $\epsilon_i \geq 2\epsilon_{i-1}$ and $\epsilon_t = O(\epsilon)$, so $\sum_{i=1}^{t} \epsilon_i = O(\epsilon)$. Moreover, our choice of ϵ_i also guarantees that $\|X_i\|_0$ is roughly $\mathrm{poly}(\epsilon^{-1}k \cdot \log^{(i)} \|X\|_0)$. Since $\log^{(i)} \|X\|_0$ decreases very fast with respect to i, $\|X_i\|_0$ becomes $\mathrm{poly}(\epsilon^{-1}k)$ in about $t = \log^\star \|X\|_0$ iterations. The detailed algorithm $\mathcal{A}'$ can be found in Algorithm 1, and we present the formal analysis in Theorem 3.1.

To construct the actual coresets which is to be used with the reduction, we adapt the importance sampling method that was proposed by Feldman and Langberg [FL11]. In previous works, the size

of the coresets from importance sampling is related to the shattering dimension of metric balls system (i.e. in our language, it is the shattering dimension of $\mathcal{F} = \{d(x,\cdot) \mid x \in X\}$.) Instead of considering the metric balls only, we give a generalized analysis where we consider a general set of "distance functions" $\mathcal{F}$ that has some error but is still "close" to d. The advantage of doing so is that we could trade the accuracy with the shattering dimension, which in turn reduces the size of the coreset.

We particularly examine two types of such functions $\mathcal{F} = \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$. The first type $\mathcal{F}$ introduces a multiplicative $(1+\epsilon)$ error to d, i.e. $\forall x \in X, c \in V$, $d(x,c) \le f_x(c) \le (1+\epsilon)\cdot d(x,c)$. Such a small distortion is already very helpful to obtain an $O(\log \|X\|_0)$ shattering dimension for minor-free graphs and Euclidean spaces. In addition to the multiplicative error, the other type of $\mathcal{F}$ introduces a certain additive error, and we make use of this to show $O(k)$ shattering dimension bound for bounded highway dimension graphs and doubling spaces. In this section, we will discuss how the two types of function sets imply efficient coresets, and the dimension bounds for various metric families will be analyzed in Section 4 where we also present the coreset results.

3.1 Iterative Size Reduction

Theorem 3.1 (Iterative Size Reduction)**.** *Let $\rho \ge 1$ be a constant and let $\mathcal{M}$ be a family of metric spaces. Assume $\mathcal{A}(X,k,z,\epsilon,\delta,M)$ is a randomized algorithm that constructs an ϵ-coreset of size $\epsilon^{-\rho}s(k)\log\delta^{-1}\log\|X\|_0$ for (k,z)-Clustering on every weighted set $X \subseteq V$ and metric space $M(V,d) \in \mathcal{M}$, for every $z \ge 1, 0 < \epsilon, \delta < \frac{1}{4}$, running in time $T(\|X\|_0,k,z,\epsilon,\delta,M)$ with success probability $1-\delta$. Then algorithm $\mathcal{A}'(X,k,z,\epsilon,\delta,M)$, stated in Algorithm 1, computes an ϵ-coreset of size $\tilde{O}(\epsilon^{-\rho}s(k)\log\delta^{-1})$ for (k,z)-Clustering on every weighted set $X \subseteq V$ and metric space $M(V,d) \in \mathcal{M}$, for every $z \ge 1$, $0 < \epsilon, \delta < \frac{1}{4}$, in time*

$$O(T(\|X\|_0,k,z,O(\epsilon/(\log\|X\|_0)^{\frac{1}{\rho}}),O(\delta/\|X\|_0),M)\cdot\log^\star\|X\|_0),$$

and with success probability $1-\delta$.

Algorithm 1 Iterative size reduction $\mathcal{A}'(X,k,z,\epsilon,\delta,M)$

Require: algorithm $\mathcal{A}(X,k,z,\epsilon,\delta,M)$ that computes an ϵ-coreset for (k,z)-Clustering on X with size $\epsilon^{-\rho}s(k)\log\delta^{-1}\log\|X\|_0$ and success probability $1-\delta$.

1: let $X_0 := X$, and let t be the largest integer such that $\log^{(t-1)}\|X\|_0 \ge \max\{20\epsilon^{-\rho}s(k)\log\delta^{-1}, \rho 2^{\rho+1}\}$
2: **for** $i = 1,\cdots,t$ **do**
3: let $\epsilon_i := \epsilon/(\log^{(i)}\|X\|_0)^{\frac{1}{\rho}}$, $\delta_i := \delta/\|X_{i-1}\|_0$
4: let $X_i := \mathcal{A}(X_{i-1},k,z,\epsilon_i,\delta_i,M)$
5: **end for**
6: $X_{t+1} := \mathcal{A}(X_t,k,z,\epsilon,\delta,M)$
7: return X_{t+1}

Proof. For the sake of presentation, let $n := \|X\|_0$, $s := s(k)$, and $\Gamma := s\epsilon^{-\rho}\log\delta^{-1}$. We start with proving in the following that X_t is an $O(\epsilon)$-coreset of X with size $\max\{160000\Gamma^4, 20\Gamma\rho^3 2^{3\rho+3}\}$ with probability $1-O(\delta)$.

Let $a_i := \|X_i\|_0$. Then by definition of X_i,

$$\begin{aligned} a_i &= s\epsilon_i^{-\rho} \log a_{i-1} \log \delta_i^{-1} \\ &= s\epsilon_i^{-\rho} \log a_{i-1} (\log a_{i-1} + \log \delta^{-1}) \\ &\le s\epsilon_i^{-\rho} \log \delta^{-1} (\log a_{i-1})^2 \end{aligned} \tag{2}$$

where the inequality is by $\log a_{i-1} + \log \delta^{-1} \le \log a_{i-1} \cdot \log \delta^{-1}$, which is equivalent to $(\log a_{i-1} - 1)(\log \delta^{-1} - 1) \ge 1$ and the latter is true because $a_{i-1} \ge \epsilon^{-\rho} \ge \epsilon^{-1} \ge 4$ and $\delta < \frac{1}{4}$.

Next we use induction to prove that $a_i \le 20\Gamma \log \delta^{-1} (\log^{(i)} n)^3$ for all $i = 1, \dots, t$. This is true for the base case when $i = 1$, since $a_1 \le s\epsilon_1^{-\rho} \log \delta^{-1} (\log n)^2 \le \Gamma (\log n)^3 \le 20\Gamma (\log n)^3$. Then consider the inductive case $i \ge 2$ and assume the hypothesis is true for $i - 1$. We have

$$\begin{aligned} a_i &\le s\epsilon_i^{-\rho} \log \delta^{-1} (\log a_{i-1})^2 && \text{by (2)} \\ &= \Gamma \log^{(i)} n \cdot (\log a_{i-1})^2 && \text{by definition of } \epsilon_i \\ &\le \Gamma \log^{(i)} n \cdot (\log(20\Gamma (\log^{(i-1)} n)^3))^2 && \text{by induction hypothesis} \\ &= \Gamma \log^{(i)} n \cdot (\log(20\Gamma) + 3 \log^{(i)} n)^2 \\ &\le \Gamma \log^{(i)} n \cdot (2(\log(20\Gamma))^2 + 18(\log^{(i)} n)^2) && \text{by } (a+b)^2 \le 2a^2 + 2b^2 \\ &\le 20\Gamma (\log^{(i)} n)^3, \end{aligned}$$

where the last inequality follows from the fact that $\log(20\Gamma) \le \log(\log^{(i-1)} n) = \log^{(i)} n$, by $i \le t$ and the definition of t. Hence we conclude $a_i \le 20\Gamma (\log^{(i)} n)^3$. This in particular implies that $a_t \le 20\Gamma (\log^{(t)} n)^3$, and by definition of t, we have $\log^{(t)} n < \max\{20\Gamma, \rho 2^{\rho+1}\}$. Hence,

$$a_t \le \max\{160000\Gamma^4, 20\Gamma\rho^3 2^{3\rho+3}\}.$$

By the guarantee of $\mathcal{A}$, we know that X_t is a $\Pi_{i=1}^t (1+\epsilon_i)$-coreset for X. Note that $a \ge 2^\rho \log a$ for every $a \ge \rho 2^{\rho+1}$, so we have $\epsilon_{i+1} \ge 2\epsilon_i$ for $i \le t$, which implies that $\sum_{i=1}^t \epsilon_i \le 2\epsilon_t$. Hence we conclude that

$$\Pi_{i=1}^t (1+\epsilon_i) \le \exp\left(\sum_{i=1}^t \epsilon_i\right) \le \exp(2\epsilon_t) \le \exp\left(\frac{2\epsilon}{(\log^{(t)} n)^{\frac{1}{\rho}}}\right) \le \exp(2\epsilon) \le 1 + 10\epsilon,$$

where the second last inequality follows from $\log^{(t)} n = \log(\log^{(t-1)} n) \ge \log(\rho 2^{\rho+1}) \ge 1$ for $\rho \ge 1$, and the last inequality follows by the fact that $\exp(2\epsilon) \le 1 + 10\epsilon$ for $\epsilon \in (0, 1)$. For the failure probability, we observe that $a_{i-1} \ge \epsilon_{i-1}^{-\rho} \ge \log^{(i-1)} n$, hence $\delta_i = \frac{\delta}{a_{i-1}} \le \frac{\delta}{\log^{(i-1)} n}$, and the total failure probability is

$$\sum_{i=1}^t \delta_i \le \delta \left(\frac{1}{n} + \frac{1}{\log n} + \cdots + \frac{1}{\log^{(t-1)} n}\right) \le O(\delta),$$

where again we have used $\log^{(t-1)} n \ge \rho 2^{\rho+1} \ge 4$, by definition of t and $\rho \ge 1$.

Therefore, X_t is an $O(\epsilon)$-coreset of X with size $\max\{160000\Gamma^4, 20\Gamma\rho^3 2^{3\rho+3}\}$ with probability $1 - O(\delta)$. Finally, in the end of algorithm $\mathcal{A}'$, we apply $\mathcal{A}$ again on X_t with parameter ϵ and δ to obtain an $O(\epsilon)$-coreset of X with size $s\epsilon^{-\rho} \log \delta^{-1} \log(\max\{160000\Gamma^4, 20\Gamma\rho^3 2^{3\rho+3}\}) = \tilde{O}(s\epsilon^{-\rho} \log \delta^{-1})$ with probability $1 - O(\delta)$.

To see the running time, we note that $t = O(\log^\star n)$, and we run $\mathcal{A}$ for $t + 1$ times. Moreover, since $\epsilon_i \ge \epsilon_1$ and $\delta_i \ge \delta_1$, the running time of each call of $\mathcal{A}$ is at most $T(\|X\|_0, k, z, \epsilon_1, \delta_1, M)$. This completes the proof of Theorem 3.1. ∎

3.2 Importance Sampling

We proceed to design the algorithm $\mathcal{A}$ required by Theorem 3.1. It is based on the importance sampling algorithm introduced by [LS10, FL11], and at a high level consists of two steps:

1. Computing probabilities: for each $x \in X$, compute $p_x \geq 0$ such that $\sum_{x \in X} p_x = 1$.
2. Sampling: draw N (to be determined later) independent samples from X, each drawn from the distribution $(p_x : x \in X)$, and assign each sample x a weight $\frac{w_X(x)}{p_x \cdot N}$ to form a coreset D.

The key observation in the analysis of this algorithm is that the sample size N, which is also the coreset size $\|D\|_0$, is related to the shattering dimension (see Definition 2.1) of a suitably defined set of functions [FL11, Theorem 4.1]. The analysis in [FL11] has been subsequently improved [BFL16, FSS20], and we make use of [FSS20, Theorem 31], restated as follows.

Lemma 3.2 (Analysis of Importance Sampling [FSS20])**.** *Fix $z \geq 1$, $0 < \epsilon < \frac{1}{2}$, an integer $k \geq 1$ and a metric space (V, d). Let $X \subseteq V$ have weights $w_X : V \to \mathbb{R}_+$ and let $\mathcal{F} := \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ be a corresponding set of functions with weights $w_{\mathcal{F}}(f_x) = w_X(x)$. Suppose $\{\sigma_x\}_{x \in X}$ satisfies*

$$\forall x \in X, \quad \sigma_x \geq \sigma_x^{\mathcal{F}} := \max_{C \in V^k} \frac{w_X(x) \cdot (f_x(C))^z}{\mathrm{cost}_z(\mathcal{F}, C)},$$

and set a suitable

$$N = O(\epsilon^{-2} \sigma_X (k \cdot \mathrm{sdim}_{\max}(\mathcal{F}) \cdot \log(\mathrm{sdim}_{\max}(\mathcal{F})) \cdot \log \sigma_X + \log \tfrac{1}{\delta})),$$

where $\sigma_X := \sum_{x \in X} \sigma_x$ and

$$\mathrm{sdim}_{\max}(\mathcal{F}) := \max_{v : X \to \mathbb{R}_+} \mathrm{sdim}\,(\mathcal{F}_v)\,, \qquad \mathcal{F}_v := \{f_x \cdot v(x) \mid x \in X\}.$$

Then the weighted set D of size $\|D\|_0 = N$ returned by the above importance sampling algorithm satisfies, with high probability $1 - \delta$,

$$\forall C \in V^k, \quad \sum_{x \in D} w_D(x) \cdot (f_x(C))^z \in (1 \pm \epsilon) \cdot \mathrm{cost}_z(\mathcal{F}, C).$$

Remark 3.3. We should explain how [FSS20, Theorem 31] implies Lemma 3.2. First of all, the bound in [FSS20] is with respect to VC-dimension, and we transfer to shattering dimension by losing a logarithmic factor (see Section 2 for the relation between VC-dimension and shattering dimension). Another main difference is that the functions therein are actually not from V to $\mathbb{R}_+$. For $\mathcal{F} = \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$, they consider $\mathcal{F}^k := \{f_x(C) = \min_{c \in C}\{f_x(c)\} \mid x \in X\}$, and their bound on the sample size is

$$N = \tilde{O}(\epsilon^{-2} \sigma_X (\mathrm{sdim}_{\max}(\mathcal{F}^k) \cdot \log \sigma_X + \log \tfrac{1}{\delta})).$$

The notion of balls and shattering dimension they use (for $\mathcal{F}^k$) is the natural extension of our Definition 2.1 (from functions on V to functions on V^k), where a ball around $C \in V^k$ is $B_{\mathcal{F}}(C, r) = \{f_x \in \mathcal{F} : f_x(C) \leq r\}$, and (1) is replaced by

$$\left|\{B_{\mathcal{H}}(C, r) : C \in V^k, r \geq 0\}\right| \leq |\mathcal{H}|^t.$$

Our Lemma 3.2 follows from [FSS20, Theorem 31] by using the fact $\mathrm{sdim}(\mathcal{F}^k) \leq k \cdot \mathrm{sdim}(\mathcal{F})$ from [FL11, Lemma 6.5].

Terminal Embeddings. As mentioned in Section 1, $\mathcal{F}$ in Lemma 3.2 corresponds to the distance function d, i.e., $f_x(\cdot) = d(x, \cdot)$, and Lemma 3.2 is usually applied directly to the distances, i.e., on a function set $\mathcal{F} = \{f_x(\cdot) = d(x, \cdot) \mid x \in X\}$. In our applications, we instead use Lemma 3.2 with a "proxy" function set $\mathcal{F}$ that is viewed as a *terminal embedding* on X, in which both the distortion of distances (between X and all of V) and the shattering dimension are controlled.

We consider two types of terminal embeddings $\mathcal{F}$. The first type (Section 3.3) maintains $(1+\epsilon)$-multiplicative distortion of the distances, and achieves dimension bound $O(\text{poly}(k/\epsilon) \log \|X\|_0)$, and the other type of $\mathcal{F}$ (Section 3.4) maintains additive distortion on top of the multiplicative one, but then the dimension is reduced to $\text{poly}(k/\epsilon)$. In what follows, we discuss how each type of terminal embedding is used to construct coresets.

3.3 Coresets via Terminal Embedding with Multiplicative Distortion

The first type of terminal embedding distorts distances between V and X multiplicatively, i.e.,

$$\forall x \in X, c \in V, \qquad d(x, c) \leq f_x(c) \leq (1 + \epsilon)\, d(x, c). \tag{3}$$

This natural guarantee works very well for (k, z)-Clustering in general. In particular, using such $\mathcal{F}$ in Lemma 3.2, our importance sampling algorithm will produce (with high probability) an $O(z\epsilon)$-coreset for (k, z)-Clustering.

Sensitivity Estimation. To compute a coreset using Lemma 3.2 we need to define, for every $x \in X$,

$$\sigma_x \geq \sigma_x^{\mathcal{F}} = \max_{C \in V^k} \frac{w_X(x) \cdot (f_x(C))^z}{\text{cost}_z(\mathcal{F}, C)}.$$

The quantity $\sigma_x^{\mathcal{F}}$, usually called the *sensitivity* of point $x \in X$ with respect to $\mathcal{F}$ [LS10, FL11]; essentially measures the maximal contribution of x to the clustering objective over all possible centers $C \subseteq V$. Since $f_x(y)$ approximates $d(x, y)$ by (3), it actually suffices to estimate the sensitivity with respect to d instead of $\mathcal{F}$, given by

$$\sigma_x^\star := \max_{C \in V^k} \frac{w_X(x) \cdot (d(x, C))^z}{\text{cost}_z(X, C)}. \tag{4}$$

Even though computing $\sigma_x^\star$ exactly seems computationally difficult, we shown next (in Lemma 3.4) that a good estimate can be efficiently computed given an $(O(1), O(1))$-approximate clustering. A weaker version of this lemma was presented in [VX12] for the case where X has unit weights, and we extend it to X with general weights. We will need the following notation. Given a subset $C \subseteq V$, denote the nearest neighbor of $x \in X$, i.e., the point in C closest to x with ties broken arbitrarily, by $\text{NN}_C(x) := \arg\min\{d(x, y) : y \in C\}$. The tie-breaking guarantees that every x has a unique nearest neighbor, and thus $\text{NN}_C(.)$ partitions X into $|C|$ subsets. The *cluster of x under C* is then defined as $C(x) := \{x' \in X : \text{NN}_C(x') = \text{NN}_C(x)\}$.

Lemma 3.4. *Fix $z \geq 1$, an integer $k \geq 1$, and a weighted set X. Given $C^{\text{apx}} \in V^k$ that is an (α, β)-approximate solution for (k, z)-Clustering on X, define for every $x \in X$,*

$$\sigma_x^{\text{apx}} := w_X(x) \cdot \left(\frac{(d(x, C^{\text{apx}}))^z}{\text{cost}_z(X, C^{\text{apx}})} + \frac{1}{w_X(C^{\text{apx}}(x))} \right).$$

Then $\sigma_x^{\text{apx}} \geq \Omega(\sigma_x^\star / (\beta 2^{2z}))$ for all $x \in X$, and $\sigma_X^{\text{apx}} := \sum_{x \in X} \sigma_x^{\text{apx}} \leq 1 + \alpha k$.

Before proving this lemma, we record the following approximate triangle inequality for distances raised to power $z \geq 1$.

Claim 3.5. *For all $x, x', y \in V$ we have $d^z(x,y) \leq 2^{z-1} \cdot [d^z(x,x') + d^z(x',y)]$.*

Proof of Claim 3.5. We first use the triangle inequality,

$$d^z(x,y) \leq [d(x,x') + d(x',y)]^z$$

and since $a \mapsto a^z$ is convex (recall $z \geq 1$), all $a, b \geq 0$ satisfy $(\frac{a+b}{2})^z \leq \frac{a^z+b^z}{2}$, hence

$$\leq 2^{z-1}[d^z(x,x') + d^z(x',y)].$$

The claim follows. □

Proof of Lemma 3.4. Given C^*, we shorten the notation by setting $\mu := \mathrm{NN}_{C^{\mathrm{apx}}}$, and let X^{apx} be the weighted set obtained by mapping all points of X by μ. Formally, $X^{\mathrm{apx}} := \{\mu(x) : x \in X\}$ where every $y \in X^{\mathrm{apx}}$ has weight $w_{X^{\mathrm{apx}}}(y) := \sum_{x \in X : \mu(x) = y} w_X(x)$. Then obviously

$$\forall x \in X, \quad w_X(C^{\mathrm{apx}}(x)) = \sum_{x' \in C^{\mathrm{apx}}(x)} w_X(x') = w_{X^{\mathrm{apx}}}(\mu(x)).$$

Upper bound on σ_X^{apx}. Using the above,

$$\sigma_X^{\mathrm{apx}} = \sum_{x \in X} \sigma_x^{\mathrm{apx}} = \sum_{x \in X} w_X(x) \cdot \left(\frac{d^z(x, \mu(x))}{\mathrm{cost}_z(X, C^{\mathrm{apx}})} + \frac{1}{w_{X^{\mathrm{apx}}}(\mu(x))} \right),$$

and we can bound

$$\sum_{x \in X} w_X(x) \cdot \frac{1}{w_{X^{\mathrm{apx}}}(\mu(x))} = \sum_{y \in X^{\mathrm{apx}}} w_{X^{\mathrm{apx}}}(y) \cdot \frac{1}{w_{X^{\mathrm{apx}}}(y)} \leq \|C^{\mathrm{apx}}\|_0 \leq \alpha k,$$

and we conclude that $\sigma_X^{\mathrm{apx}} \leq 1 + \alpha k$, as required.

Lower bound on σ_x^{apx} (relative to $\sigma_x^\star$). Aiming to prove this as an upper bound on $\sigma_x^\star$, consider for now a fixed $C \in V^k$. We first establish the following inequality, that relates the cost of X^{apx} to that of X.

$$\begin{aligned}
\mathrm{cost}_z(X^{\mathrm{apx}}, C) &= \sum_{y \in X^{\mathrm{apx}}} w_{X^{\mathrm{apx}}}(y) \cdot d^z(y, C) \\
&= \sum_{x \in X} w_X(x) \cdot d^z(\mu(x), C) \\
&\leq 2^{z-1} \sum_{x \in X} w_X(x) \cdot [d^z(\mu(x), x) + d^z(x, C)] && \text{by Claim 3.5} \\
&= 2^{z-1} \cdot [\mathrm{cost}_z(X, C^{\mathrm{apx}}) + \mathrm{cost}_z(X, C)] && \text{as } C^{\mathrm{apx}} \text{ is } (\alpha, \beta)\text{-approximation} \\
&\leq 2^{z-1}(\beta + 1) \cdot \mathrm{cost}_z(X, C). && (5)
\end{aligned}$$

Now aiming at an upper bound on $\sigma_x^\star$, observe that

$$\frac{d^z(X, C)}{\mathrm{cost}_z(X, C)} \leq 2^{z-1} \cdot \left[\frac{d^z(x, \mu(x)) + d^z(\mu(x), C)}{\mathrm{cost}_z(X, C)} \right] \qquad \text{by Claim 3.5} \qquad (6)$$

and let us bound each term separately. For the first term, since C^{apx} is an (α, β)-approximation,

$$\frac{d^z(x,\mu(x))}{\text{cost}_z(X,C)} \le \beta \cdot \frac{d^z(x,\mu(x))}{\text{cost}_z(X,C^{\text{apx}})}.$$

The second term is

$$\begin{aligned}
\frac{d^z(\mu(x),C)}{\text{cost}_z(X,C)} &\le (\beta+1)2^{z-1} \cdot \frac{d^z(\mu(x),C)}{\text{cost}_z(X^{\text{apx}},C)} && \text{by (5)} \\
&= (\beta+1)2^{z-1} \cdot \frac{d^z(\mu(x),C)}{\sum_{y\in X^{\text{apx}}} w_{X^{\text{apx}}}(y)\cdot d^z(y,C)} \\
&\le (\beta+1)2^{z-1} \cdot \frac{1}{w_{X^{\text{apx}}}(\mu(x))}.
\end{aligned}$$

Plugging these two bounds into (6), we obtain

$$\frac{d^z(x,C)}{\text{cost}_z(X,C)} \le (\beta+1)2^{2z-2} \cdot \left[\frac{d^z(x,\mu(x))}{\text{cost}_z(X,C^{\text{apx}})} + \frac{1}{w_{X^{\text{apx}}}(\mu(x))}\right] = (\beta+1)2^{2z-2} \cdot \frac{\sigma_x^{\text{apx}}}{w_X(x)}.$$

Using the definition in (4), we conclude that $(\beta+1)2^{2z-2}\cdot\sigma_x^{\text{apx}} \ge \sigma_x^\star$, which completes the proof of Lemma 3.4. □

Conclusion. Our importance sampling algorithm for this type of terminal embedding is listed in Algorithm 2. By a direct combination of Lemma 3.2 and Lemma 3.4, we conclude that the algorithm yields a coreset, which is stated formally in Lemma 3.6.

Algorithm 2 Coresets for (k,z)-Clustering for $\mathcal{F}$ with multiplicative distortion

1: compute an $(O(1),O(1))$-approximate solution C^{apx} for (k,z)-Clustering on X
2: for each $x \in X$, let $\sigma_x := w_X(x)\cdot\left(\frac{(d(x,C^{\text{apx}}))^z}{\text{cost}_z(X,C^{\text{apx}})} + \frac{1}{w_X(C^{\text{apx}}(x))}\right)$ ▷ as in Lemma 3.4
3: for each $x \in X$, let $p_x := \frac{\sigma_x}{\sum_{y\in X}\sigma_y}$
4: draw $N := O\left(\epsilon^{-2}2^{2z}k\cdot\left(zk\log k\cdot\text{sdim}_{\max}(\mathcal{F}) + \log\frac{1}{\delta}\right)\right)$ independent samples from X, each from the distribution $(p_x : x \in X)$ ▷ $\text{sdim}_{\max}$ as in Lemma 3.2
5: let D be the set of samples, and assign each $x \in D$ a weight $w_D(x) := \frac{w_X(x)}{p_x N}$
6: return the weighted set D

Lemma 3.6. *Fix $0 < \epsilon, \delta < \frac{1}{2}$, $z \ge 1$, an integer $k \ge 1$, and a metric space $M(V,d)$. Given a weighted set $X \subseteq V$ and respective $\mathcal{F} = \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ such that*

$$\forall x \in X, c \in V, \qquad d(x,c) \le f_x(c) \le (1+\epsilon)\cdot d(x,c),$$

Algorithm 2 computes a weighted set $D \subseteq X$ of size

$$\|D\|_0 = O\left(\epsilon^{-2}2^{2z}k\left(zk\log k\cdot\text{sdim}_{\max}(\mathcal{F}) + \log\tfrac{1}{\delta}\right)\right),$$

that with high probability $1-\delta$ is an ϵ-coreset for (k,z)-Clustering on X.

The running time of Algorithm 2 is dominated by the sensitivity estimation, especially line 1 which computes an $(O(1),O(1))$-approximate solution. In Lemma 3.7 we present efficient implementations of the algorithm, both in metric settings and in graph settings.

Lemma 3.7. *Algorithm 2 can be implemented in time $\tilde{O}(k\|X\|_0)$ if it is given oracle access to the distance d, and it can be implemented in time $\tilde{O}(|E|)$ if the input is an edge-weighted graph $G = (V, E)$ and M is its shortest-path metric.*

Proof. The running time is dominated by Step 1 which requires an $(O(1), O(1))$-approximation in both settings. For the metric setting where oracle access to d is given, [MP04] gave an $\tilde{O}(k\|X\|_0)$ algorithm for both k-MEDIAN ($z = 1$) and k-MEANS ($z = 2$), and it has been observed to work for general z in a recent work [HV20].

For the graph setting, Thorup [Tho05, Theorem 20] gave an $(2, 12 + o(1))$-approximation for graph k-MEDIAN in time $\tilde{O}(|E|)$, such that the input points are *unweighted*. Even though not stated in his result, we observe that his approach may be easily modified to handle weighted inputs as well, and we briefly mention the major changes.

- Thorup's first step [Tho05, Algorithm D] is to compute an $(\tilde{O}(\log |V|), O(1))$-approximation F by successive uniform independent sampling. This can be naturally modified to sampling proportional to the weights of the input points.
- Then, the idea is to use the Jain-Vazirani algorithm [JV01] on the bipartite graph $F \times X$. To make sure the running time is $\tilde{O}(|V|)$, the edges of $F \times X$ sub-sampled by picking, for each $x \in X$, only $\tilde{O}(1)$ neighbors in F. This sampling is oblivious to weights, and hence still goes through. Let the sampled subgraph be G'.
- Finally, the Jain-Vazirani algorithm is applied on G' to obtain the final $(2, 12 + o(1))$-approximation. However, we still need to modify Jain-Vazirani to work with weighted inputs. Roughly, Jain-Vazirani algorithm is a primal-dual method, so the weights are easily incorporated to the linear program, and the primal-dual algorithm is naturally modified so that dual variables are increased at a rate that is proportional to their weight in the linear program.

After obtaining C^{apx}, the remaining steps of Algorithm 2 trivially runs in time $\tilde{O}(k\|X\|_0)$ when oracle access to d is given. However, for the graph setting, the trivial implementation of Step 2 which requires to compute $\text{cost}_1(X, C^{\text{apx}})$ needs to run $\tilde{O}(k)$ single-source-shortest-paths from points in C^{apx}, and this leads to a running time $\tilde{O}(k|V|)$. In fact, as observed in [Tho05, Observation 1], only one single-source-shortest-path needs to be computed, by running Dijkstra's algorithm on a virtual point x_0 which connects to each point in C^{apx} to x_0 with 0 weight.

This completes the proof of Lemma 3.7. □

3.4 Coresets via Terminal Embedding with Additive Distortion

The second type of embedding has, in addition to the above $(1 + \epsilon)$-multiplicative distortion, also an additive distortion. Specifically, we assume the function set $\mathcal{F} = \mathcal{F}_S$ is defined with respect to some subset $S \subseteq V$ and satisfies

$$\forall x \in X, c \in V, \quad d(x, c) \le f_x(c) \le (1 + \epsilon) \cdot d(x, c) + \epsilon \cdot d(x, S).$$

The choice of S clearly affects the dimension $\text{sdim}_{\max}(\mathcal{F}_S)$, but let us focus now on the effect on the clustering objective, restricting our attention henceforth only to the case $z = 1$ (recalling that $\text{cost}_1 = \text{cost}$). Suppose we pick $S := C^{\text{apx}}$ where C^{apx} is an (α, β)-approximation for k-MEDIAN. Then even though the additive error for any given x, y might be very large, it will preserve the k-MEDIAN objective for X, because

$$\begin{aligned} \forall C \in V^k, \quad \text{cost}(X, C) \le \text{cost}(\mathcal{F}, C) &\le (1 + \epsilon) \cdot \text{cost}(X, C) + \epsilon \cdot \text{cost}(X, C^{\text{apx}}) \\ &\le (1 + (\beta + 1)\epsilon) \cdot \text{cost}(X, C). \end{aligned} \tag{7}$$

However, this does not immediately imply a coreset for k-MEDIAN, because we need an analogous bound, but for D instead of X (recall that D is computed by importance sampling with respect to $\mathcal{F}$). In particular, using Lemma 3.2 and (7) we get one direction (with high probability)

$$\forall C \in V^k, \quad \sum_{x \in D} w_D(x) \cdot f_x(C) \geq (1-\epsilon) \cdot \mathrm{cost}(\mathcal{F}, C) \geq (1-\epsilon) \geq \mathrm{cost}(X, C),$$

however in the other direction we only have

$$\forall C \in V^k, \quad \sum_{x \in D} w_D(x) \cdot f_x(C) \leq (1+\epsilon) \cdot \mathrm{cost}(D, X) + \epsilon \cdot \sum_{x \in D} w_D(x) \cdot d(x, C^{\mathrm{apx}}),$$

where the term $\sum_{x \in D} w_D(x) \cdot d(x, C^{\mathrm{apx}})$ remains to be bounded.

This term $\sum_{x \in D} w_D(x) \cdot d(x, C^{\mathrm{apx}})$ can be viewed as a weak coreset guarantee which preserves the objective $\mathrm{cost}(X, \cdot)$ on C^{apx} only. Fortunately, because C^{apx} is fixed before the importance sampling, our algorithm may be interpreted as estimating a fixed sum

$$\mathrm{cost}(X, C^{\mathrm{apx}}) = \sum_{x \in X} w_X(x) \cdot d(x, C^{\mathrm{apx}})$$

using independent samples in D, i.e., by the estimator $\sum_{x \in D} w_D(x) \cdot d(x, C^{\mathrm{apx}})$. And now Hoeffding's inequality shows that for large enough N, this estimator is accurate with high probability.

We present our new algorithm in Algorithm 3, which is largely similar to Algorithm 2, except for a slightly larger number of samples N and some hidden constants. Hence, its running time is similar to Algorithm 2, as stated in Corollary 3.8 for completeness. Its correctness requires new analysis and is presented in Lemma 3.9.

Algorithm 3 Coresets for k-MEDIAN on $\mathcal{F}$ with additive distortion

1: compute an $(O(1), O(1))$-approximate solution C^{apx} for k-MEDIAN on X
2: for each $x \in X$, let $\sigma_x^{\mathrm{apx}} := w_X(x) \cdot \left(\frac{d(x, C^{\mathrm{apx}})}{\mathrm{cost}(X, C^{\mathrm{apx}})} + \frac{1}{w_X(C^{\mathrm{apx}}(x))} \right)$ ▷ as in Lemma 3.4
3: for each $x \in X$, let $p_x := \frac{\sigma_x^{\mathrm{apx}}}{\sum_{y \in X} \sigma_y^{\mathrm{apx}}}$
4: draw $N := O\left(\epsilon^{-2} k \left(k \log k \cdot \mathrm{sdim}_{\max}(\mathcal{F}_{C^{\mathrm{apx}}}) + \log \frac{1}{\delta} \right) + k^2 \log \frac{1}{\delta} \right)$ independent samples from X, each from the distribution $(p_x : x \in X)$ ▷ $\mathrm{sdim}_{\max}$ as in Lemma 3.2, and $\mathcal{F}_{C^{\mathrm{apx}}}$ as in (8)
5: for each x in the sample D assign weight $w_D(x) := \frac{w_X(x)}{p_x N}$
6: return the weighted set D

Corollary 3.8. *Algorithm 3 can be implemented in time $\tilde{O}(k \|X\|_0)$ if it is given oracle access to the distance d, and in time $\tilde{O}(|V| + |E|)$ if the input is an edge-weighted graph $G = (V, E)$ and M is its shortest-path metric.*

Lemma 3.9. *Fix $0 < \epsilon, \delta < \frac{1}{2}$, an integer $k \geq 1$, and a metric space $M(V, d)$. Given a weighted set $X \subseteq V$, and an $(O(1), O(1))$-approximate solution $C^{\mathrm{apx}} \in V^k$ for k-MEDIAN on X, suppose $\mathcal{F}_{C^{\mathrm{apx}}} = \{ f_x : V \to \mathbb{R}_+ \mid x \in X \}$ satisfies*

$$\forall x \in X, c \in V, \quad d(x, c) \leq f_x(c) \leq (1+\epsilon) \cdot d(x, c) + \epsilon \cdot d(x, C^{\mathrm{apx}}); \tag{8}$$

then Algorithm 3 computes a weighted set $D \subseteq X$ of size

$$\|D\|_0 = O\left(\epsilon^{-2} k \left(k \log k \cdot \mathrm{sdim}_{\max}(\mathcal{F}_{C^{\mathrm{apx}}}) + \log \tfrac{1}{\delta} \right) + k^2 \log \tfrac{1}{\delta} \right),$$

that with high probability $1 - \delta$ is an ϵ-coreset for k-MEDIAN on X.

Proof. Suppose $C^{\mathrm{apx}} \in V^k$ is an (α, β)-approximate solution for $\alpha, \beta = O(1)$. Observe that (8) implies (7), and write $\mathcal{F} = \mathcal{F}_{C^{\mathrm{apx}}}$ for brevity.

Sensitivity Analysis. We would like to employ Lemma 3.2. Observe that σ_x^{apx} in Algorithm 3 is the same, up to hidden constants, as in Algorithm 2, hence the upper bound $\sigma_X^{\text{apx}} \leq 1 + \alpha k$ follows immediately from Lemma 3.4. We also need to prove that $\sigma_x^{\text{apx}} \geq \Omega(\sigma_x^{\mathcal{F}})$ for all $x \in X$, where $\sigma_x^{\mathcal{F}} = \max_{C \in V^k} \frac{w_X(x) \cdot f_x(C)}{\text{cost}(\mathcal{F}, C)}$. Once again, we aim to prove this as an upper bound on $\sigma_x^{\mathcal{F}}$.

Fix $x \in X$, and let $C^{\max} \in V^k$ be a maximizer in the definition of $\sigma_x^{\mathcal{F}}$ (which clearly depends on x). Then

$$\begin{aligned}
\sigma_x^{\mathcal{F}} &= \frac{w_X(x) \cdot f_x(C^{\max})}{\text{cost}(\mathcal{F}, C^{\max})} \\
&\leq \frac{w_X(x) \cdot [(1+\epsilon) \cdot d(x, C^{\max}) + \epsilon \cdot d(x, C^{\text{apx}})]}{\text{cost}(X, C^{\max})} && \text{by (8) and (7)} \\
&\leq (1+\epsilon) \cdot \sigma_x^{\star} + \epsilon \cdot \frac{w_X(x) \cdot d(x, C^{\text{apx}})}{\text{cost}(X, C^{\max})} && \text{as defined in (4)} \\
&\leq (1+\epsilon) \cdot \sigma_x^{\star} + \beta\epsilon \cdot \frac{w_X(x) \cdot d(x, C^{\text{apx}})}{\text{cost}(X, C^{\text{apx}})} && \text{as } C^{\text{apx}} \text{ is } (\alpha, \beta)\text{-approximation} \\
&\leq (1+\epsilon) \cdot \sigma_x^{\star} + \beta\epsilon \cdot \sigma_x^{\text{apx}}. && \text{as defined in line 2}
\end{aligned}$$

Combining this with our bound $\sigma_x^{\star} \leq O(\beta) \cdot \sigma_x^{\text{apx}}$ from Lemma 3.4 (recall $z = 1$), we conclude that $\sigma_x^{\mathcal{F}} \leq O(\beta) \cdot \sigma^{\text{apx}}$.

Overall Error Bound. Recall our goal is to prove that with probably at least $1 - \delta$, the output D is a coreset, i.e.,

$$\forall C \in V^k, \quad \text{cost}(D, C) \in (1 \pm O(\beta\epsilon)) \cdot \text{cost}(X, C). \tag{9}$$

Applying Lemma 3.2 with our choice of N in line 4 of the algorithm, we know that with probability at least $1 - \delta/2$,

$$\forall C \in V^k, \quad \sum_{x \in D} w_D(x) \cdot f_x(C) \in (1 \pm \epsilon) \cdot \text{cost}(\mathcal{F}, C) \tag{10}$$

We claim, and will prove shortly, that with probability at least $1 - \delta/2$,

$$\sum_{x \in D} w_D(x) \cdot d(x, C^{\text{apx}}) \leq 2 \cdot \text{cost}(X, C^{\text{apx}}). \tag{11}$$

Using this claim, we complete the proof as follows. By a union bound, with probability at least $1 - \delta$, both (10) and (11) hold. In this case, for all $C \in V^k$, one direction of (9) follows easily

$$\begin{aligned}
\text{cost}(D, C) &\leq \sum_{x \in D} w_D(x) \cdot f_x(C) && \text{by (8)} \\
&\leq (1+\epsilon) \cdot \text{cost}(\mathcal{F}, C) && \text{by (10)} \\
&\leq (1 + O((\beta\epsilon)) \cdot \text{cost}(X, C). && \text{by (7)}
\end{aligned}$$

For the other direction of (9), which crucially rely on (11), we have

$$\begin{aligned}
\operatorname{cost}(X,C) &\le \operatorname{cost}(\mathcal{F},C) && \text{by (7)}\\
&\le \frac{1}{1-\epsilon}\cdot\sum_{x\in D} w_D(x)\cdot f_x(C) && \text{by (10)}\\
&\le \frac{1+\epsilon}{1-\epsilon}\cdot\operatorname{cost}(D,C)+\frac{\epsilon}{1-\epsilon}\cdot\sum_{x\in D} w_D(x)\cdot d(x,C^{\mathrm{apx}}) && \text{by (8)}\\
&\le \frac{1+\epsilon}{1-\epsilon}\cdot\operatorname{cost}(D,C)+\frac{2\epsilon}{1-\epsilon}\cdot\operatorname{cost}(X,C^{\mathrm{apx}}), && \text{by (11)}
\end{aligned}$$

and finally using that C^{apx} is (α,β)-approximation and some rearrangement, we get that $\operatorname{cost}(X,C)\le(1+O(\beta\epsilon))\operatorname{cost}(D,C)$.

It remains to prove our claim, i.e., that (11) holds with high probability. This follows by a straightforward application of Hoeffding's Inequality. To see this, define for each $1\le i\le N$ the random variable $Y_i := \frac{w_X(x)\cdot d(x,C^{\mathrm{apx}})}{p_x}$, where x is the i-th sample in line 4, and let $Y := \frac{1}{N}\sum_{i=1}^{N} Y_i$. Then

$$Y=\sum_{x\in D} w_D(x)\cdot d(x,C^{\mathrm{apx}}),$$

and its expectation is $\mathbb{E}[Y]=\mathbb{E}[Y_1]=\sum_{x\in X} w_X(x)\cdot d(x,C^{\mathrm{apx}})=\operatorname{cost}(X,C^{\mathrm{apx}})$.

Now observe that the random variables Y_i are independent, and use Lemma 3.4 to bound each of them by

$$0\le Y_i=\frac{w_X(x)\cdot d(x,C^{\mathrm{apx}})}{\sigma_x^{\mathrm{apx}}/\sigma_X^{\mathrm{apx}}}\le(1+\alpha k)\cdot\operatorname{cost}(x,C^{\mathrm{apx}})=(1+\alpha k)\,\mathbb{E}[Y].$$

Hence, by Hoeffding's Inequality

$$\forall t>0,\quad \Pr\left[Y-\mathbb{E}[Y]>t\right]\le\exp\left(-\frac{2Nt^2}{((1+\alpha k)\,\mathbb{E}[Y])^2}\right)$$

and for $t=\mathbb{E}[Y]$ and a suitable $N\ge\Omega(\alpha^2k^2\log\frac{1}{\delta})$, we conclude that $\Pr\left[Y>2\,\mathbb{E}[Y]\right]\le\delta/2$. This proves the claim and completes the proof of Lemma 3.9. □

4 Coresets

We now apply the framework developed in Section 3 to design coresets of size independent of X for various settings, including excluded-minor graphs (in Section 4.1), high-dimensional Euclidean spaces (in Section 4.3), and graphs with bounded highway dimension (in Section 4.4). Our workhorse will be Lemma 3.6 and Lemma 3.9, which effectively translate a terminal embedding $\mathcal{F}$ with low distortion on $X\times V$ and low shattering dimension $\operatorname{sdim}_{\max}$ into an efficient algorithm to construct a coreset whose size is linear in $\operatorname{sdim}_{\max}(\mathcal{F})$.

We therefore turn our attention to designing various terminal embeddings. For excluded-minor graphs, we design a terminal embedding $\mathcal{F}$ with multiplicative distortion $1+\epsilon$ of the distances, and dimension $\operatorname{sdim}_{\max}(\mathcal{F})=O(\operatorname{poly}(k/\epsilon)\cdot\log\|X\|_0)$. For Euclidean spaces, we employ a known terminal embedding with similar guarantees. In both settings, even though the shattering dimension depends on $\|X\|_0$, it still implies coresets of size independent of X by our iterative size reduction (Theorem 3.1). We thus obtain the first coreset (of size independent of X and V) for excluded-minor graphs (Corollary 4.2), and a simpler state-of-the-art coreset for Euclidean spaces (Corollary 4.18).

We also design a terminal embedding for graphs with bounded highway dimension (formally defined in Section 4.4). This embedding has an additive distortion (on top of the multiplicative one), but its shattering dimension is independent of X, hence the iterative size reduction is not required. We thus obtain the first coreset (of size independent of X and V) for graphs with bounded highway dimension (Corollary 4.25).

4.1 Excluded-minor Graphs

Our terminal embedding for excluded-minor graphs is stated in the next lemma. Previously, the shattering dimension of the shortest-path metric of graphs excluding a fixed graph H_0 as a minor was studied only for unit point weight, for which Bousquet and Thomassé [BT15] proved that $\mathcal{F} = \{d(x,\cdot) \mid x \in X\}$ has shattering dimension $\mathrm{sdim}(\mathcal{F}) = O(|H_0|)$. For arbitrary point weight, i.e., $\mathrm{sdim}_{\max}(\mathcal{F})$, it is still open to get a bound that depends only on $|H_0|$, although the special case of bounded treewidth was recently resolved, as Baker et al. [BBH+20], proved that $\mathrm{sdim}_{\max}(\mathcal{F}) = O(\mathrm{tw}(G))$ where $\mathrm{tw}(G)$ denotes the treewidth of the graph G. Note that both of these results use no distortion of the distances, i.e., they bound $\mathcal{F} = \{d(x,\cdot) \mid x \in X\}$. Our terminal embedding handles the most general setting of excluded-minor graphs and arbitrary point weight, although it bypasses the open question by allowing a small distortion and dependence on X.

Lemma 4.1 (Terminal Embedding for Excluded-minor Graphs)**.** *For every edge-weighted graph $G = (V, E)$ that excludes some fixed minor and whose shortest-path metric is denoted as $M = (V, d)$, and for every weighted set $X \subseteq V$, there exists a set of functions $\mathcal{F} := \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ such that*

$$\forall x \in X, c \in V, \qquad d(x,c) \le f_x(c) \le (1+\epsilon) \cdot d(x,c),$$

and $\mathrm{sdim}_{\max}(\mathcal{F}) = \tilde{O}(\epsilon^{-2}) \cdot \log \|X\|_0$.

Let us present now an overview of the proof of Lemma 4.1, deferring the full details to Section 4.2. Our starting point is the following approach, which was developed in [BBH+20] for bounded-treewidth graphs. (The main purpose is to explain how vertex separators are used as portals to bound the shattering dimension, but unfortunately additional technical details are needed.) The first step in this approach reduces the task of bounding the shattering dimension to counting how many distinct permutations of X one can obtain by ordering the points of X according to their distance from a point c, when ranging over all $c \in V$. An additional argument uses the bounded treewidth to reduce the range of c from all of V to a subset $\hat{V} \subset V$, that is separated from X by a vertex-cut $P \subset V$ of size $|\hat{P}| = O(1)$. This means that every path, including the shortest-path, between every $x \in X$ and every $c \in \hat{V}$ must pass through $\hat{P}$, therefore

$$d(x,c) = \min\{d(x,p) + d(p,c) : p \in \hat{P}\},$$

and the possible orderings of X are completely determined by these values. The key idea now is to replace the hard-to-control range of $c \in \hat{V}$ with a richer but easier range of $|\hat{P}| = O(1)$ real variables. Indeed, each $d(x,\cdot)$ is captured by a *min-linear function*, which means a function of the form $\min_i a_i y_i + b_i$ with real variables $\{y_i\}$ that represent $\{d(p,c)\}_{p \in \hat{P}}$ and fixed coefficients $\{a_i, b_i\}$. Therefore, each $d(x,\cdot)$ is captured by a min-linear function $g_x : \mathbb{R}^{|\hat{P}|} \to \mathbb{R}_+$, and these functions are all defined on the same $|\hat{P}| = O(1)$ real variables. In this representation, it is easy to handle the point weight $v : X \to \mathbb{R}_+$ (to scale all distances from x), because each resulting function $v(x) \cdot g_x$ is still min-linear. Finally, the number of orderings of the set $\{g_x\}_{x \in X}$ of min-linear functions, is counted using the arrangement number for hyperplanes, which is a well-studied quantity in computational geometry.

To extend this approach to excluded-minor graphs (or even planar graphs), which do not admit small vertex separators, we have to replace vertex separators with shortest-path separators [Tho04, AG06]. In particular, we use these separator theorem to partition the whole graph into a few parts, such that each part is separated from the graph by only a few shortest paths, see Lemma 4.5 for planar graphs (which is a variant of a result known from [EKM14]) and Lemma 4.12 for excluded-minor graphs. However, the immediate obstacle is that while these separators consist of a few paths, their total size is unbounded (with respect to X), which breaks the above approach because each min-linear function has too many variables. A standard technique to address this size issue is to discretize the path separator into *portals*, and reroute through them a shortest-path from each $x \in X$ to each $c \in V$. This step distorts the distances, and to keep the distortion bounded multiplicatively by $1+\epsilon$, one usually finds inside each separating shortest-path l, a set of portals $P_l \subset l$ whose spacing is at most $\epsilon \cdot d(x,c)$. However, $d(x,c)$ could be very small compared to the entire path l, hence we cannot control the number of portals (even for one path l).

Vertex-dependent Portals In fact, all we need is to represent the relative ordering of $\{d(x,\cdot) : x \in X\}$ using a set of *min-linear functions* over a few real variables, and these variables do not have to be the distance to *fixed portals* on the separating shortest paths. (Recall this description is eventually used by the arrangement number of hyperplanes to count orderings of X.) To achieve this, we first define *vertex-dependent* portals P_c^l with respect to a separating shortest path l *and* a vertex $c \in V$ (notice this includes also P_x^l for $x \in X$). and then a shortest path from $x \in X$ to $c \in V$ passing through l is rerouted through portals $P_x^l \cup P_c^l$, as follows. First, since l is itself a shortest path, $d(x,c) = \min_{u_1,u_2 \in l}\{d(x,u_1)+d(u_1,u_2)+d(u_2,c)\}$. Observe that $d(u_1,u_2)$ is already linear, because one real variable can "capture" a location in l, hence we only need to approximate $d(x,u_1)$ and $d(c,u_2)$. To do so, we approximate the distances from c to every vertex on the path l, i.e., $\{d(c,u)\}_{u \in l}$, using only the distances from c to its portal set P_c^l, i.e., $\{d(c,p)\}_{p \in P_c^l}$. Moreover, between successive portals this approximate distance is a linear function, and it actually suffices to use $|P_c^l| = \text{poly}(1/\epsilon)$ portals, which means that $d(c,u)$ can be represented as a *piece-wise linear* function in $\text{poly}(1/\epsilon)$ real variables.

Note that the above approach ends up with the minimum of piece-wise linear (rather than linear) functions, which creates extra difficulty. In particular, we care about the relative ordering of $\{d(x,\cdot) : x \in X\}$ over all $c \in V$, and to evaluate $d(x,c)$ we need the pieces that c and x generate, i.e., information about $P_c^l \cup P_x^l$. Since the number of $c \in V$ is unbounded, we need to "guess" the structure of P_c^l, specifically the ordering between the portals in P_c^l and those in P_x^l. Fortunately, since every $|P_c^l| \le \text{poly}(1/\epsilon)$, such a "guess" is still affordable, and this would prove Lemma 4.1.

Corollary 4.2 (Coresets for Excluded-Minor Graphs)**.** *For every edge-weighted graph $G=(V,E)$ that excludes a fixed minor, every $0 < \epsilon, \delta < 1/2$ and integer $k \ge 1$, k-Median of every weighted set $X \subseteq V$ (with respect to the shortest path metric of G) admits an ϵ-coreset of size $\tilde{O}(\epsilon^{-4}k^2 \log \frac{1}{\delta})$. Furthermore, such a coreset can be computed in time $\tilde{O}(|E|)$ with success probability $1-\delta$.*

Proof. By combining Lemma 3.6, Lemma 3.7 with our terminal embedding from Lemma 4.1, we obtain an efficient algorithm for constructing a coreset of size $\tilde{O}(\epsilon^{-4}k^2 \log \|X\|_0)$. This size can be reduced to the claimed size (and running time) using the iterative size reduction of Theorem 3.1. □

Remark 4.3. This result partly extends to (k,z)-Clustering for all $z \ge 1$. The importance sampling algorithm and its analysis are immediate, and in particular imply the existence of a coreset of size $\tilde{O}(\epsilon^{-4}k^2 \log \frac{1}{\delta})$. However we rely on known algorithm for $z=1$ in the step of computing an approximate clustering (needed to compute sampling probabilities).

4.2 Proof of Lemma 4.1

For the sake of presentation, we start with proving the planar case, since this already requires most of our new technical ideas. The statement of terminal embedding for planar graphs is as follows, and how the proof can be modified to work for the minor-excluded case is discussed in Section 4.2.1.

Lemma 4.4 (Terminal Embedding for Planar Graphs)**.** *For every edge-weighted planar graph $G = (V, E)$ whose shortest path metric is denoted as $M = (V, d)$ and every weighted set $X \subseteq V$, there exists a set of functions $\mathcal{F} = \mathcal{F}_X := \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ such that for every $x \in X$, and $c \in V$, $f_x(c) \in (1 \pm \epsilon) \cdot d(x, c)$, and $\mathrm{sdim}_{\max}(\mathcal{F}) = \widetilde{O}(\epsilon^{-2}) \log \|X\|_0$.*

By definition, $\mathrm{sdim}_{\max}(\mathcal{F}) = \max_{v:X\to\mathbb{R}_+}(\mathcal{F}_v)$, so it suffices to bound $\mathrm{sdim}(\mathcal{F}_v)$ for every v. Also, by the definition of sdim, it suffices to prove for every $\mathcal{H} \subseteq \mathcal{F}_v$ with $|\mathcal{H}| \geq 2$,

$$|\{B_{\mathcal{H}}(c, r) : c \in V, r \geq 0\}| \leq \mathrm{poly}(\|X\|_0) \cdot |\mathcal{H}|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}.$$

Hence, we fix some $v : X \to \mathbb{R}_+$ and $\mathcal{H} \subseteq \mathcal{F}_v$ with $|\mathcal{H}| \geq 2$ throughout the proof.

General Reduction: Counting Relative Orderings For $\mathcal{H} \subseteq \mathcal{F}$ and $c \in V$, let $\sigma_c^{\mathcal{H}}$ be the permutation of $\mathcal{H}$ ordered by $v(x) \cdot f_x(c)$ in non-decreasing order and ties are broken arbitrarily. Then for a fixed $c \in V$ and very $r \geq 0$, the subset $B_{\mathcal{H}}(c, r) \subseteq \mathcal{H}$ is exactly the subset defined by some prefix of $\sigma_c^{\mathcal{H}}$. Hence,

$$|\{B_{\mathcal{H}}(c, r) : c \in V, r \geq 0\}| \leq |\mathcal{H}| \cdot \left|\{\sigma_c^{\mathcal{H}} : c \in V\}\right|.$$

Therefore, it suffices to show

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V\}\right| \leq \mathrm{poly}(\|X\|_0) \cdot |\mathcal{H}|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}.$$

Hence, this reduces the task of bounding of shattering dimension to counting the number of relative orderings of $\{v(x) \cdot f_x(c) \mid x \in X\}$.

Next, we use the following structural lemma for planar graphs to break the graph into few parts of simple structure, so we can bound the number of permutations for c coming from each part. A variant of this lemma has been proved in [EKM14], where the key idea is to use the *interdigitating* trees. For completeness, we give a full proof of this lemma in Appendix A.

Lemma 4.5 (Structural Property of Planar Graphs, see also [EKM14])**.** *For every edge-weighted planar graph $G = (V, E)$ and subset $S \subseteq V$, V can be broken into parts $\Pi := \{V_i\}_i$ with $|\Pi| = \mathrm{poly}(|S|)$ and $\bigcup_i V_i = V$, such that for every $V_i \in \Pi$,*

1. *$|S \cap V_i| = O(1)$,*
2. *there exists a collection of shortest paths $\mathcal{P}_i$ in G with $|\mathcal{P}_i| = O(1)$ and removing the vertices of all paths in $\mathcal{P}_i$ disconnects V_i from $V \setminus V_i$ (points in V_i are possibly removed).*

Furthermore, such Π and the corresponding shortest paths $\mathcal{P}_i$ for $V_i \in \Pi$ can be computed in $\tilde{O}(|V|)$ time[7]*.*

[7] This lemma is used only in the analysis in this section, but the running time is relevant when this lemma is used again in Section 5.

Applying Lemma 4.5 with $S = X$ (noting that S is an unweighted set), we obtain $\Pi = \{V_i\}_i$ with $|\Pi| = \text{poly}(\|X\|_0)$, such that each part $V_i \in \Pi$ is separated by $O(1)$ shortest paths $\mathcal{P}_i$. Then

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V\}\right| \le \sum_{V_i \in \Pi} \left|\{\sigma_c^{\mathcal{H}} : c \in V_i\}\right|.$$

Hence it suffices to show for every $V_i \in \Pi$, it holds that

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V_i\}\right| \le |\mathcal{H}|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}. \tag{12}$$

Since $\bigcup_i V_i = V$, it suffices to define functions $f_x(\cdot)$ for $c \in V_i$ for every i independently. Therefore, we fix $V_i \in \Pi$ throughout the proof. In the following, our proof proceeds in three parts. The first defines functions $f_x(\cdot)$ on V_i, the second analyzes the distortion of f_x's, and the final part analyzes the shattering dimension.

Part I: Definition of f_x on V_i By Lemma 4.5 we know $|V_i \cap X| = O(1)$. Hence, the "simple" case is when $x \in V_i \cap T$, for which we define $f_x(\cdot) := d(x, \cdot)$.

Otherwise, $x \in X \setminus V_i$. Write $\mathcal{P}_i := \{P_j\}_j$. Since P_j's are shortest paths in G, and removing $\mathcal{P}_i$ from G disconnects V_i from $V \setminus V_i$, we have the following fact.

Fact 4.6. *For $c \in V_i$ and $x \in X \setminus V_i$, there exists $P_j \in \mathcal{P}_i$ and $c', x' \in P_j$, such that $d(c, x) = d(c, c') + d(c', x') + d(x', x)$.*

Let $d_j(c, x)$ be the length of the shortest path from c to x that uses *at least one* point in P_j. For each $P_j \in \mathcal{P}_i$, we will define $f_x^j : V_i \to \mathbb{R}_+$, such that $f_x^j(c)$ is within $(1 \pm \epsilon) \cdot d_j(c, x)$, and let

$$f_x(c) := \min_{P_j \in \mathcal{P}_i} f_x^j(c), \qquad \forall c \in V_i.$$

Hence, by Fact 4.6, the guarantee that $f_x^j(c) \in (1 \pm \epsilon) \cdot d_j(c, x)$ implies $f_x(c) \in (1 \pm \epsilon) \cdot d(x, c)$, as desired. Hence we focus on defining f_x^j in the following.

Defining $f_x^j : V_i \to \mathbb{R}_+$ Suppose we fix some $P_j \in \mathcal{P}_i$, and we will define $f_x^j(c)$, for $c \in V_i$. By Fact 4.6 and the optimality of shortest paths, we have

$$d_j(x, c) = \min_{c', x' \in P_j} \{d(c, c') + d(c', x') + d(x', x)\}.$$

For every $y \in V$, we will define $l_y^j : P_j \to \mathbb{R}_+$ such that $l_y^j(y') \in (1 \pm \epsilon) \cdot d(y, y')$ for every $y' \in P_j$. Then, we let

$$f_x^j(c) := \min_{c', x' \in P_j} \{l_c^j(c') + d(c', x') + l_x^j(x')\},$$

and this would imply $f_x^j(c) \in (1 \pm \epsilon) \cdot d_j(x, c)$. So it remains to define $l_y^j : P_j \to \mathbb{R}_+$ for every $y \in V$.

Defining $l_y^j : P_j \to \mathbb{R}_+$ Fix $y \in V$ and we will define $l_y^j(y')$ for every $y' \in P_j$. Pick $h_y \in P_j$ that satisfies $d(y, h_y) = d(y, P_j)$. Since P_j is a shortest path, we interpret P_j as a segment in the real line. In particular, we let the two end points of P_j be 0 and 1, and P_j is a (discrete) subset of $[0, 1]$.

Define $a, b \in P_j$ such that $a \le h_y \le b$ are the two *furthest* points on the two sides of h on P_j that satisfy $d(h_y, a) \le \frac{d(y, h_y)}{\epsilon}$ and $d(h_y, b) \le \frac{d(y, h_y)}{\epsilon}$. Then construct a sequence of points $a = q_1 \le q_2 \dots$ in the following way. For $t = 1, 2, \dots$, if there exists $u \in (q_t, 1] \cap P_j$ such that $d(q_t, u) > \epsilon \cdot d(y, h_y)$, then let q_{t+1} be the smallest such u; if such u does not exist, then let $q_{t+1} := b$ and terminate. Essentially, this breaks P_j into segments of length $\epsilon \cdot d(y, h_y)$, except that the last one that ends with b may be shorter. Denote this sequence as $Q_y := (q_1 = a, \dots, q_m = b)$.

Claim 4.7. *For every $y \in V$, $|Q_y| = O(\epsilon^{-2})$.*

Proof. By the definition of Q_y, for $1 \le t \le m-2$, $d(q_t, q_{t+1}) > \epsilon \cdot d(y, h_y)$. On the other hand, by the definition of a and b, $d(q_1, q_m) = d(a, b) \le O(\frac{d(y,h_y)}{\epsilon})$. Therefore, $|Q_y| \le O(\epsilon^{-2})$, as desired. □

Definition of f_x on V_i: Recap Define

$$l_y^j(y') := \begin{cases} d(h_y, y') & \text{if } y' < a = q_1 \text{ or } y' > b = q_m \\ d(y, q_t) & \text{if } q_t \le y' < q_{t+1}, 1 \le t < m \\ d(y, q_m) & \text{if } y' = b = q_m \end{cases} \tag{13}$$

where $h_y \in P_j$, $Q_y = \{q_t\}_t \subset P_j$. To recap,

- if $x \in X \cap V_i$, then $f_x(c) := d(x, c)$;
- otherwise $x \in X \setminus V_i$, $f_x(c) := \min_{P_j \in \mathcal{P}_i} f_x^j(c)$, where

$$f_x^j(c) := \min_{c', x' \in P_j} \{l_c^j(c') + d(c', x') + l_x^j(x')\}. \tag{14}$$

Finally,

$$f_x(c) := \min_{P_j \in \mathcal{P}_i} f_x^j(c), \qquad \forall c \in V_i. \tag{15}$$

Part II: Distortion Analysis The distortion of l's is analyzed in the following Lemma 4.8, and the distortion for f_x follows immediately from the above definitions.

Lemma 4.8. *For every $P_j \in \mathcal{P}_i$, $y \in V$, $y' \in P_j$, $l_y^j(y') \in (1 \pm \epsilon) \cdot d(y, y')$.*

Proof. If $y' = q_m = b$, by definition $l_y^j(y') = d(y, q_m) = d(y, y')$. Then consider the case when $y' < a = q_1$ or $y' > b = q_m$.

$$\begin{aligned} l_y^j(y') &= d(h_y, y') \\ &\in d(y', y) \pm d(y, h_y) \\ &\in d(y', y) \pm \epsilon \cdot d(y', h_y), \end{aligned}$$

where the last inequality follows from $d(y', h_y) > \frac{d(y,h_y)}{\epsilon}$. This implies $d(y, y') \in (1 \pm \epsilon) \cdot l_y^j(y')$.

Otherwise, $q_t \le y' < q_{t+1}$ for some $1 \le t < m$. By the definition of q_t's and the definition of h_y,

$$\begin{aligned} d(y, y') &\in d(y, q_t) \pm d(q_t, y') \\ &\in d(y, q_t) \pm \epsilon \cdot d(y, h_y) \\ &\in d(y, q_t) \pm \epsilon \cdot d(y, y') \\ &\in l_y^j(y') \pm \epsilon \cdot d(y, y'), \end{aligned}$$

which implies $l_y^j(y') \in (1 \pm \epsilon) \cdot d(y, y')$. This finishes the proof of Lemma 4.8. □

Part III: Shattering Dimension Analysis Recall that we fixed $v : X \to \mathbb{R}_+$ and $\mathcal{H} \subseteq \mathcal{F}_v$ with $|\mathcal{H}| \geq 2$. Now we show

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V_i\}\right| \leq |\mathcal{H}|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}. \tag{16}$$

Let $H := \{x : v(x) \cdot f_x \in \mathcal{H}\}$, so $|H| = |\mathcal{H}|$. Recall that $|V_i \cap X| = O(1)$ by Lemma 4.5, so $|V_i \cap H| = O(1)$. Hence, if we could show

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V_i\}\right| \leq N(|H|)$$

for $\mathcal{H}$ such that $H \cap V_i = \emptyset$, then for general $\mathcal{H}$,

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V_i\}\right| \leq N(|H| - |V_i \cap H|) \cdot |H|^{O(|V_i \cap H|)} \leq N(|H|) \cdot |H|^{O(1)}.$$

Therefore, it suffices to show (16) under the assumption that $H \cap V_i = \emptyset$.

In the following, we will further break V_i into $|H|^{\tilde{O}(\epsilon^{-2})}$ parts, such that for each part V', f_x on V' may be alternatively represented as a *min-linear* function.

Lemma 4.9. *Let $u = |\mathcal{P}_i|$. There exists a partition Γ of V_i, such that the following holds.*

1. $|\Gamma| \leq |H|^{\tilde{O}(\epsilon^{-2}) \cdot u}$.
2. $\forall V' \in \Gamma$, $\forall x \in H$, *there exists* $g_x : \mathbb{R}^s \to \mathbb{R}_+$ *where* $s = O(\epsilon^{-2})$, *such that* g_x *is a minimum of* $O(\epsilon^{-4}u)$ *linear functions on* $\mathbb{R}^s$, *and for every* $c \in V'$, *there exists* $y \in \mathbb{R}^s$ *that satisfies* $f_x(c) = g_x(y)$.

Proof. Before we actually prove the lemma, we need to examine $f_x^j(c)$ and l_y^j more closely. Suppose some $P_j \in \mathcal{P}_i$ is fixed. Recall that for $y \in V, y' \in P_j$ (defined in (13)),

$$l_y^j(y') := \begin{cases} d(h_y, y') & \text{if } y' < a = q_1 \text{ or } y' > b = q_m \\ d(y, q_t) & \text{if } q_t \leq y' < q_{t+1}, 1 \leq t < m \\ d(y, q_m) & \text{if } y' = b = q_m \end{cases}$$

where $h_y \in P_j$, $Q_y = \{q_t\}_t \subset P_j$. Hence, for every y, l_y^j is a *piece-wise linear* function with $O(|Q_y|) = O(\epsilon^{-2})$ (by Claim 4.7) pieces, where the transition points of l_y^j are $Q_y \cup \{0, 1\}$ (noting that $d(h_y, y')$ is linear since $h_y, y' \in P_j$).

Using that l's are piece-wise linear, we know for $c \in V_i, x \in X \setminus V_i$,

$$\begin{aligned} f_x^j(c) &= \min_{c', x' \in P_j} \{l_c^j(c') + d(c', x') + l_x^j(x')\} && \text{defined in (14)} \\ &= \min_{c', x' \in Q_c \cup Q_x \cup \{0,1\}} \{l_c^j(c') + d(c', x') + l_x^j(x')\}. && \text{as } l\text{'s are piece-wise linear} \end{aligned}$$

Hence, to evaluate $f_x^j(c)$ we only need to evaluate $l_c^j(c')$ and $l_x^j(x')$ at $c', x' \in Q_c \cup Q_x \cup \{0, 1\}$, and in particular we need to find the piece in l_c^j and l_x^j that every $c', x' \in Q_c \cup Q_x \cup \{0, 1\}$ belong to, and then evaluate a linear function. Precisely, the piece that every c', x' belongs to is determined by the relative ordering of points $Q_x \cup Q_c$ (recalling that they are from P_j). Thus, the pieces are not only determined by x, but also by c which is the variable, and this means without the information about the pieces, f_x cannot be represented as a min-linear function g_x. Therefore, the idea is to find a partition Γ of V_i, such that for c in each part $V' \in \Gamma$, the relative ordering of Q_c with respect to $\{Q_x : x \in H\}$ is the same. We note that we need to consider the ordering of Q_c with respect to all Q_x's, because we care about the relative orderings of all f_x's.

Defining Γ For $1 \le j \le u$, $c \in V_i$, let τ_c^j be the ordering of Q_c with respect to $\bigcup_{y\in H} Q_y$ on P_j. Here, an ordering of Q_c with respect to $\left(\bigcup_{y\in H} Q_y\right)$ is defined by their ordering on P_j which is interpreted as the real line. In our definition of Γ, we will require each part $V' \in \Gamma$ to satisfy that $\forall c \in V'$, the tuple of orderings $(\tau_c^1, \ldots, \tau_c^u)$ remains the same. That is, V_i is partitioned according to the joint relative ordering τ_c^j's on all shortest paths $P_j \in \mathcal{P}_i$.

Formally, for $1 \le j \le u$, let $\Lambda^j := \{\tau_c^j : c \in V_i\}$ be the collection of distinct ordering τ_c^j on P_j over points $c \in V_i$. Define

$$\Lambda := \Lambda^1 \times \ldots \times \Lambda^u$$

as the tuples of τ_j's for $1 \le j \le u$ (here, the $\times$ operator is the Cartesian product). For $(\tau_1, \ldots, \tau_u) \in \Lambda$, define

$$V_i^{(\tau_1,\ldots,\tau_u)} := \{c \in V_i : (\tau_c^1 = \tau_1) \wedge \ldots \wedge (\tau_c^u = \tau_u)\}$$

as the subset of V_i such that the ordering τ_c^j for each $1 \le j \le u$ agrees with the given tuple. Finally, we define the partition as

$$\Gamma := \{V_i^{(\tau_1,\ldots,\tau_u)} : (\tau_1, \ldots, \tau_u) \in \Lambda\}.$$

Bounding $|\Gamma|$ By Claim 4.7, we know $|Q_y| = O(\epsilon^{-2})$ for every $y \in V$. Hence, $\left|\bigcup_{y\in H} Q_y\right| = O\left(\epsilon^{-2}|H|\right)$. Therefore, for every $j \in [u]$,

$$|\Lambda^j| \le \binom{O(\epsilon^{-2}|H|)}{O(\epsilon^{-2})} = O\left(\epsilon^{-1}|H|\right)^{O(\epsilon^{-2})}.$$

Therefore,

$$|\Gamma| \le \Pi_{1\le j\le u}|\Lambda^j| \le O\left(\epsilon^{-1}|H|\right)^{O(\epsilon^{-2}u)} \le |H|^{\tilde{O}(\epsilon^{-2})\cdot u},$$

as desired.

Defining g_x By our definition of Γ, we need to define g_x for each $V' \in \Gamma$. Now, fix tuple $(\tau_1, \ldots, \tau_u) \in \Lambda$, so the part corresponds to this tuple is $V' = V_i^{(\tau_1,\ldots,\tau_u)}$, and we will define g_x with respect to such V'. Similar to the definition of f_x's (see (15)), we define $g_x : \mathbb{R}^s \to \mathbb{R}_+$ to have the form

$$g_x(y) := \min_{P_j\in\mathcal{P}_i} g_x^j(y).$$

Then, for $1 \le j \le u$, $x \in H$, define $g_x^j : \mathbb{R}^s \to \mathbb{R}$ of $s := O(\epsilon^{-2})$ variables $(q_1, \ldots, q_m, d(c, q_1), \ldots, d(c, q_m), h_c)$ for $q_i \in Q_c$, such that

$$g_x^j(q_1, \ldots, q_m, d(c, q_1), \ldots, d(c, q_m), h_c) = \min_{c',x'\in Q_c\cup Q_x\cup\{0,1\}} \{l_c^j(c') + d(c', x') + l_x^j(x')\}.$$

We argue that for every $1 \le j \le u$, g_x^j may be viewed as a minimum of $O(\epsilon^{-4})$ linear functions whose variables are the same with that of g_x^j.

- Linearity. Suppose $c \in V'$, and fix $c', x' \in Q_c \cup Q_x \cup \{0,1\}$. By the above discussions, $l_c^j(c')$ could take values only from $\{d(c,q_i) : q_i \in Q_c\} \cup \{d(h_c,c')\}$. Since $\forall q_i \in Q_c$, $d(c,q_i)$ is a variable of g_x^j, and $d(h_c,c') = |h_c - c'|$ is linear and that h_c is also a variable of g_x^j, we conclude that $l_c^j(c')$ may be written as a linear function of the same set of variables of g_x^j. By a similar argument, we have the same conclusion for l_x^j. Therefore, $l_c^j(c') + d(c',x') + l_x^j(x')$ may be written as a linear function of $(q_1,\ldots,q_m,d(c,q_1),\ldots,d(c,q_m),h_c)$.
- Number of linear functions. By Claim 4.7, we have

$$\forall y \in V, \qquad |Q_y| = O(\epsilon^{-2}),$$

hence $|Q_c \cup Q_x \cup \{0,1\}| = O(\epsilon^{-2})$. Therefore, there are $O(\epsilon^{-4})$ pairs of $c', x' \in Q_c \cup Q_x \cup \{0,1\}$.

Therefore, item 2 of Lemma 4.9 follows by combining this with the definition of g_x. We completed the proof of Lemma 4.9. □

Now suppose Γ is the one that is guaranteed by Lemma 4.9. Since

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V_i\}\right| \le \sum_{V' \in \Gamma} \left|\{\sigma_c^{\mathcal{H}} : c \in V'\}\right|$$

and

$$|\Gamma| \le |H|^{\tilde{O}(\epsilon^{-2})\cdot u} \le |H|^{\tilde{O}(\epsilon^{-2})}, \tag{17}$$

where the last inequality is by Lemma 4.5 (recalling $u = |\mathcal{P}_i|$), it suffices to show for every $V' \in \Gamma$,

$$\left|\{\sigma_c^{\mathcal{H}} : c \in V'\}\right| \le |H|^{\tilde{O}(\epsilon^{-2})\log\|X\|_0}. \tag{18}$$

Fix some $V' \in \Gamma$. By Lemma 4.9, for every $x \in H$ there exists a min-linear function $g_x : \mathbb{R}^s \to \mathbb{R}_+$ ($s = O(\epsilon^{-2})$)), such that for every $c \in V'$, there exists $y \in \mathbb{R}^s$ that satisfies $f_x(c) = g_x(y)$. For $y \in \mathbb{R}^s$ define π_y^H as a permutation of H that is ordered by $g_x(y)$ in non-increasing order and ties are broken in a way that is consistent with σ. Then

$$\left|\{\sigma_c^{\mathcal{H}_v} : c \in V'\}\right| \le \left|\{\pi_y^H : y \in \mathbb{R}^s\}\right|. \tag{19}$$

We make use of the following lemma to bound the number of permutations π_y^H. The lemma relates the number of relative orderings of g_x's to the arrangement number in computational geometry.

Lemma 4.10 (Complexity of Min-linear Functions [BBH+20]). *Suppose there are m functions $g_1,\ldots,g_m$ from $\mathbb{R}^s$ to $\mathbb{R}$, such that $\forall i \in [m]$, g_i is of the form*

$$g_i(x) := \min_{j\in[t]}\{g_{ij}(x)\},$$

where g_{ij} is a linear function. For $x \in \mathbb{R}^s$, let π_x be the permutation of $[m]$ ordered by $g_i(x)$. Then,

$$|\{\pi_x : x \in \mathbb{R}^s\}| \le (mt)^{O(s)}.$$

Applying Lemma 4.10 on g_x's for $x \in H$ with parameters $s = O(\epsilon^{-2})$, $t = O(\epsilon^{-4}u) = O\left(\epsilon^{-4}\log\|X\|_0\right)$ and $m = |H|$, we obtain

$$\left|\{\pi_y^H : y \in \mathbb{R}^s\}\right| \le O\left(\epsilon^{-1}|H|\log\|X\|_0\right)^{O(\epsilon^{-2})} \le |H|^{\tilde{O}(\epsilon^{-2})\cdot\log\|X\|_0}. \tag{20}$$

Thus, (18) is implied by combining (20) with (19). Finally, we complete the proof of Lemma 4.4 by combining the above three parts of the arguments.

4.2.1 From Planar to Minor-excluded Graphs

The strategy for proving the minor-excluded case is similar to the planar case. Hence, we focus on presenting the major steps and highlight the differences, while omitting repetitive arguments. The terminal embedding lemma that we need to prove is restated as follows.

Lemma 4.11 (Restatement of Lemma 4.1)**.** *For every edge-weighted graph $G = (V, E)$ whose shortest path metric is denoted as $M = (V, d)$, and every weighted set $X \subseteq V$, given that G excludes some fixed minor, there exists a set of functions $\mathcal{F} := \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ such that for every $x \in X$, and $c \in V$, $d(x, c) \le f_x(c) \le (1+\epsilon) \cdot d(x, c)$, and $\mathrm{sdim}_{\max}(\mathcal{F}) = \tilde{O}(\epsilon^{-2}) \cdot \log \|X\|_0$.*

Similar to the planar case, we fix $v : X \to \mathbb{R}_+$ and $\mathcal{H} \subseteq \mathcal{F}_v$ with $|\mathcal{H}| \ge 2$ throughout the proof. Then σ_c^H is defined the same as before, and it suffices to prove

$$|\{\sigma_c^{\mathcal{H}} : c \in V\}| \le \mathrm{poly}(\|X\|_0) \cdot |\mathcal{H}|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}.$$

Next, we used a structural lemma to break V into several parts where each part is separated by a few shortest paths. In the planar case, we showed in Lemma 4.5 that the number of parts is $O(\|X\|_0)$, and only $O(1)$ separating shortest paths in G are necessary. However, the proof of Lemma 4.5 heavily relies on planarity, and for minor-excluded graphs, we only manage to prove the following weaker guarantee.

Lemma 4.12 (Structural Property of Minor-excluded Graphs)**.** *Given edge-weighted graph $G = (V, E)$ that excludes a fixed minor, and a subset $S \subseteq V$, there is a collection $\Pi := \{V_i\}_i$ of V with $|\Pi| = \mathrm{poly}(|S|)$ and $\bigcup_i V_i = V$ such that for every $V_i \in \Pi$ the following holds.*

1. *$|S \cap V_i| = O(1)$.*
2. *There exists an integer t_i and t_i groups of paths $\mathcal{P}_1^i, \ldots, \mathcal{P}_{t_i}^i$ in G, such that*
   (a) *$|\bigcup_{j=1}^{t_i} \mathcal{P}_j^i| = O(\log |S|)$*
   (b) *removing the vertices of all paths in $\bigcup_{j=1}^{t_i} \mathcal{P}_j^i$ disconnects V_i from $V \setminus V_i$ in G (possibly removing points in V_i)*
   (c) *for $1 \le j \le t_i$, let G_j^i be the sub-graph of G formed by removing all paths in $\mathcal{P}_1^i, \ldots, \mathcal{P}_{j-1}^i$ (define $G_1^i = G$), then every path in $\mathcal{P}_j^i$ is a shortest path in G_j^i.*

The lemma follows from a recursive application of the balanced shortest path separator theorem in [AG06, Theorem 1], stated as follows.

Lemma 4.13 (Balanced Shortest Path Separator [AG06])**.** *Given edge-weighted graph $G = (V, E)$ that excludes a fixed minor with non-negative vertex weight[8], there is a set of vertices $S \subseteq V$, such that*

1. *$S = P_1 \cup P_2 \cup \ldots$ where P_i is a set of shortest paths in the graph formed by removing $\bigcup_{j<i} P_j$*
2. *$\sum_i |P_i| = O(1)$, where the hidden constant depends on the size of the excluded minor*
3. *the weight of every component in the graph formed by removing S from G is at most half the weight of V.*

[8] [AG06, Theorem 1] only states the special case with unit vertex weight, while the general weighted version was discussed in a note of the same paper.

Proof of Lemma 4.12. Without loss of generality, we assume G is a connected graph. We will apply Lemma 4.13 on G recursively to define the partition Π and the groups of shortest paths $\{\mathcal{P}_j^i\}_j$ associated with the parts. The detailed procedure, called DEF-Π, is defined in Algorithm 4. We assume there is a global Γ initialized as $\Gamma = \emptyset$ which is constructed throughout the execution of the recursive algorithm. The execution of the algorithm starts with DEF-$\Pi(G, \emptyset, S)$.

Roughly, the procedure DEF-Π takes a sub-graph G', a set $\mathsf{sep} = \{\mathcal{P}_j\}_j$ of groups of paths and S as input, such that G' corresponds to a component in a graph formed by removing all paths in sep from G. The procedure execute on such G' and find shortest paths in G' using Lemma 4.13. The found shortest paths are segmented (with respect to S) and added to the collection Π. Then the found shortest paths are removed from G' to form a new graph G''. Components in G'' that contain less than 2 points in S are made new parts in Π, and the procedure DEF-Π is invoked recursively on other components in G''.

Algorithm 4 Procedure DEF-$\Pi(G' = (V', E'), \mathsf{sep}, S)$

1: apply Lemma 4.13 on graph G' with vertex weight 1 if $x \in V' \cap S$ and 0 otherwise, and let $\mathcal{P}$ be the set of shortest paths in G' guaranteed by the lemma.
2: **for** $P \in \mathcal{P}$ **do**
3: interpret P as interval $[0, 1]$, list $S \cap P = \{x_1, \ldots, x_m\}$ and $0 \le x_1 \le \ldots \le x_m \le 1$
4: segment P into sub-paths $\mathcal{P}' = \{[0, x_1], [x_1, x_2], \ldots, [x_m, 1]\}$
5: **for** $P' \in \mathcal{P}'$ **do**
6: include P' in Π, and define the set of associated groups of shortest paths as $\mathsf{sep} \cup \{P'\}$
7: **end for**
8: **end for**
9: let G'' be the graph formed by removing all paths in $\mathcal{P}$, and let $\mathcal{C} = \{C_i\}_i$ be its components
10: include the union of all components with no intersection with S as a single part in Π, and define the set of associated groups of paths as $\mathsf{sep} \cup \mathcal{P}$
11: **for** $C_i \in \mathcal{C}$ **do**
12: **if** $|C_i \cap S| = 1$ **then**
13: include C_i as a new part in Π, and define the set of associated groups of paths as $\mathsf{sep} \cup \mathcal{P}$
14: **else if** $|C_i \cap S| \ge 2$ **then**
15: call DEF-$\Pi(G''[C_i], \mathsf{sep} \cup \{\mathcal{P}\}, S)$ ▷ $G''[C_i]$ is the induced sub-graph of G'' on vertex set C_i
16: **end if**
17: **end for**

By construction and Lemma 4.13, it is immediate that $\bigcup_{V_i \in \Pi} V_i = V$, and item 2.(b), 2.(c) also follows easily. To see item 1, we observe that we have two types of V_i's in Π. One is from the shortest paths $\mathcal{P}$ (Line 6), and because of the segmentation, the intersection with S is at most 2. The other type is the components in G'' whose intersection with S is by definition at most 1 (Line 10, 13). Therefore, it remains to upper bound $|\Pi|$, and show item 2.(a) which requires a bound of $|\bigcup_{j=1}^{t_i} \mathcal{P}_j^i| = O(\log |S|)$ for all $V_i \in \Pi$.

First, we observe that at any execution of Gen-Π, it is always the case that $0 \le |\mathsf{sep}| \le O(\log |S|)$, because Lemma 4.13 guarantees the weight of every component in G'' is halved. This also implies that the total number of executions of GEN-Π is poly$(|S|)$. Therefore, $\forall V_i \in \Pi$, $|\bigcup_{j=1}^{t_i} \mathcal{P}_j^i| \le O(\log |S|)$, which proves item 2.(a).

Bounding $|\Pi|$ Observe that there are three places where we include a part V_i in Π, and we let Π_1 be the subset of those included at Line 6, Π_2 be those included at Line 10, and Π_3 be those

included at Line 13. Then $|\Pi| \le |\Pi_1| + |\Pi_2| + |\Pi_3|$.

If $V_i \in \Pi_1$, then V_i is a sub-path of some $P \in \mathcal{P}$, where $\mathcal{P}$ is defined at Line 1. We observe that the number of all $V_i \in \Pi_1$ such that $V_i \cap S \neq \emptyset$, i.e. $|\{V_i \in \Pi_1 : V_i \cap S \neq \emptyset\}|$, is at most $O(|S|)$. This is because we remove paths $P \in \mathcal{P}$ in every recursion, which means any point in S can only participate in at most one such P during the whole execution, and hence any point in S can intersect at most two sub-paths $V_i \in \Pi_1$ such that $V_i \cap S \neq \emptyset$ (because $|V_i \cap S| \le 2$ by the segmentation at Line 4). On the other hand, if $V_i \in \Pi_1$ and $V_i \cap S = \emptyset$, then no segmentation was performed and $V_i = P$ for P at Line 2. Therefore, the number of such V_i's is bounded by the total number of execution of DEF-Π multiplied by the size of $\mathcal{P}$ at Line 2, which is at most $\mathrm{poly}(|S|)$. Therefore, we conclude that $|\Pi_1| = \mathrm{poly}(|S|)$.

Finally, since every $V_i \in \Pi_3$ satisfies $|V_i \cap S| = 1$ (at Line 12 and 13), and we observe that subsets in Π_3 are disjoint, so we immediately have $|\Pi_3| = O(|S|)$. For Π_2, we note that only one $V_i \in \Pi_2$ could be included in each execution of DEF-Π, so $|\Pi_2| = \mathrm{poly}(|S|)$.

We conclude the proof of Lemma 4.12 by combining all the above discussions. □

As before, we still apply the Lemma 4.12 with $S = X$ (which is unweighted set) to obtain $\Gamma = \{V_i\}_i$ with $|\Pi| = O(\mathrm{poly}(\|X\|_0))$, and it suffices to prove for each $V_i \in \Pi$

$$|\{\sigma_c^{\mathcal{H}} : c \in V_i\}| \le |\mathcal{H}|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}.$$

To proceed, we fix V_i and define functions $f_x(\cdot)$ for $c \in V_i$. However, compared with Lemma 4.5, the separating shortest paths in Lemma 4.12 are not from the original graph G, but is inside some sub-graph generated by removing various other separating shortest paths. Also, the number of shortest paths in the separator is increased from $O(1)$ to $O(\log \|X\|_0)$.

Hence, we need to define f_x's with respect to the new structure of the separating shortest paths. Suppose $\{\mathcal{P}_1^i, \dots, \mathcal{P}_{t_i}^i\}$ is the t_i groups of paths guaranteed by Lemma 4.12. Also as in the lemma, suppose G_j^i is the sub-graph of G formed by removing all paths in $\mathcal{P}_1^i, \dots, \mathcal{P}_{j-1}^i$ (define $G_1^i = G$). For $1 \le j \le t_i$, $P \in \mathcal{P}_j^i$ and $x, y \in V$, let $d_j^P(x, y)$ denote the length of the shortest path from x to y using edges in G_j^i and uses at least one point of P. Then, analogue to Fact 4.6, we have the following lemma.

Lemma 4.14. *For $c \in V_i$ and $x \in V \setminus V_i$, there exists $1 \le j \le t_i$, $P \in \mathcal{P}_j^i$ and $c', x' \in P$, such that $d(c, x) = d_j^P(c, c') + d_j^P(c', x') + d_j^P(x', x)$.*

Proof. First, we observe that the shortest path $c \rightsquigarrow x$ has to intersect (at a vertex of) at least one path contained in $\{\mathcal{P}_j^i\}_j$, because removing $\bigcup_{j=1}^{t_i} \mathcal{P}_j$ disconnects V_i from $V \setminus V_i$. Suppose j_0 is the smallest j such that $c \rightsquigarrow x$ intersects a shortest path in $\mathcal{P}_j^i$, and let $P \in \mathcal{P}_{j_0}^i$ be any intersected path in $\mathcal{P}_{j_0}^i$.

Then, this implies that (the edge set of) $c \rightsquigarrow x$ is totally contained in sub-graph $G_{j_0}^i$, since $G_{j_0}^i$ is formed by removing only groups $\mathcal{P}_j^s$ with $j < j_0$ which do not intersect $c \rightsquigarrow x$. Hence, we have $d(c, x) = d_{G_{j_0}^i}(c, x)$, where $d_{G_{j_0}^i}$ is the shortest path metric in sub-graph $G_{j_0}^i$. By Lemma 4.12, P is a shortest path in $G_{j_0}^i$, so $c \rightsquigarrow x$ has to cross P at most once, which implies there exists $c', x' \in P$, such that $d(x, c) = d_j^P(c, c') + d_j^P(c', x') + d_j^P(x', x)$, as desired. □

Using Lemma 4.14 and by the optimality of the shortest path, we conclude that

$$\forall c \in V_i, x \in X, \quad d(c, x) = \min_{1 \le j \le t_i} \min_{P \in \mathcal{P}_j^i} \min_{c', x' \in P} \{d_j^P(c, c') + d_j^P(c', x') + d_j^P(x', x)\}.$$

Then, for each $1 \le j \le t_i$, path $P \in \mathcal{P}_j^i$, we use the same way as in the planar case to define the approximate distance function l to approximate $d_j^P(y, y')$ for $y \in V$ and $y' \in P$. The f_x is then defined similarly, and the distortion follows by a very similar argument as in Lemma 4.8.

The analysis of shattering dimension is also largely the same as before, except that the definition of u in the statement of Lemma 4.9 is slightly changed because of the new structural lemma. The new statement is presented as follows, and the proof of it is essentially as before.

Lemma 4.15. *Let $u = |\bigcup_{j=1}^{t_i} \mathcal{P}_j^i|$. There exists a partition Γ of V_i, such that the following holds.*

1. *$|\Gamma| \le |H|^{\tilde{O}(\epsilon^{-2}) \cdot u}$.*
2. *$\forall V' \in \Gamma$, $\forall x \in H$, there exists $g_x : \mathbb{R}^s \to \mathbb{R}_+$ where $s = O(\epsilon^{-2})$, such that g_x is a minimum of $O(\epsilon^{-4}u)$ linear functions on $\mathbb{R}^s$, and for every $c \in V'$, there exists $y \in \mathbb{R}^s$ that satisfies $f_x(c) = g_x(y)$.*

We apply the lemma with the new bound of $u = |\bigcup_{j=1}^{t_i} \mathcal{P}_j^i| = O(\log \|X\|_0)$ (by Lemma 4.12), and the bound in (18) is increased to

$$|\Gamma| \le |H|^{\tilde{O}(\epsilon^{-2}) \cdot u} \le |H|^{\tilde{O}(\epsilon^{-2}) \log \|X\|_0}.$$

Finally, to complete the proof of Lemma 4.1, we again use Lemma 4.10 on each $V' \in \Gamma$ to conclude the desired shattering dimension bound.

4.3 High-Dimensional Euclidean Spaces

We present a terminal embedding for Euclidean spaces, with a guarantee that is similar to that of excluded-minor graphs. For these results, the ambient metric space (V, d) of all possible centers is replaced by a Euclidean space.[9]

Lemma 4.16. *For every $\epsilon \in (0, 1/2)$ and finite weighted set $X \subset \mathbb{R}^m$, there exists $\mathcal{F} = \{f_x : \mathbb{R}^m \to \mathbb{R}_+ \mid x \in X\}$ such that*

$$\forall x \in X, c \in \mathbb{R}^m, \qquad \|x - c\|_2 \le f_x(c) \le (1 + \epsilon)\|x - c\|_2,$$

and $\mathrm{sdim}_{\max}(\mathcal{F}) = O(\epsilon^{-2} \log \|X\|_0)$.

Proof. The lemma follows immediately from the following terminal version of the Johnson-Lindenstrauss Lemma [JL84], proved recently by Narayanan and Nelson [NN19].

Theorem 4.17 (Terminal Johnson-Lindenstrauss Lemma [NN19])**.** *For every $\epsilon \in (0, 1/2)$ and finite $S \subset \mathbb{R}^m$, there is an embedding $g : S \to \mathbb{R}^t$ for $t = O(\epsilon^{-2} \log |S|)$, such that*

$$\forall x \in S, y \in \mathbb{R}^m, \quad \|x - y\|_2 \le \|g(x) - g(y)\|_2 \le (1 + \epsilon)\|x - y\|_2.$$

Given $X \subset \mathbb{R}^m$, apply Theorem 4.17 with $S = X$ (as an unweighted set), and define for every $x \in X$ the function $f_x(c) := \|g(x) - g(c)\|_2$. Then $\mathcal{F} = \{f_x \mid x \in X\}$ clearly satisfies the distortion bound. The dimension bound follows by plugging $t = O(\epsilon^{-2} \log \|X\|_0)$ into the bound $\mathrm{sdim}_{\max}(\mathcal{F}) = O(t)$ known from [FL11, Lemma 16.3].[10] ∎

[9] It is easily verified that as long as X is finite, our entire framework from Section 3 extends to $V = \mathbb{R}^m$ with ℓ_2 norm. For example, all maximums (e.g., in Lemma 3.2) are well-defined by using compactness arguments on a bounding box.

[10] The following is proved in [FL11, Lemma 16.3]. For every $S \subset \mathbb{R}^t$, the function set $\mathcal{H} := \{h_x \mid x \in S\}$ given by $h_x(y) = \|x - y\|_2$, has shattering dimension $\mathrm{sdim}_{\max}(\mathcal{H}) = O(t)$.

Corollary 4.18 (Coresets for Euclidean Spaces)**.** *For every $0 < \epsilon, \delta < 1/2$, $z \geq 1$, and integers $k, m \geq 1$, Euclidean (k, z)-*Clustering *of every weighted set $X \subset \mathbb{R}^m$ admits an ϵ-coreset of size $\tilde{O}(\epsilon^{-4} 2^{2z} k^2 \log \frac{1}{\delta})$. Furthermore, such a coreset can be computed*[11] *in time $\tilde{O}(k \|X\|_0 m)$ with success probability $1 - \delta$.*

Proof. By combining Lemma 3.6, Lemma 3.7 with our terminal embedding from Lemma 4.16, we obtain an efficient algorithm for constructing a coreset of size $\tilde{O}(\epsilon^{-4} 2^{2z} k^2 \log \|X\|_0)$. This size can be reduced to the claimed size (and running time) using the iterative size reduction of Theorem 3.1. □

Remark 4.19 (Comparison to [HV20])*.* For (k, z)-Clustering in Euclidean spaces, our algorithms can also compute an ϵ-coreset of size $\tilde{O}(\epsilon^{-O(z)} k)$, which offers a different parameters tradeoff than Corollary 4.18. This alternative bound is obtained by simply replacing the application of Lemma 3.2 (which is actually from [FSS20]) with [HV20, Lemma 3.1] (which itself is a result from [FL11], extended to weighted inputs).

Our two coreset size bounds are identical to the state-of-the-art bounds proved by Huang and Vishnoi [HV20] (in the asymptotic sense). Their analysis is different, and bounds $\mathrm{sdim}_{\max}$ independently of X using a dimensionality-reduction argument for clustering objectives. In contrast, we require only a loose bound $\mathrm{sdim}_{\max}(\mathcal{F}) = O(\mathrm{poly}(\epsilon^{-1}) \cdot \log \|X\|_0)$, which follows immediately from [NN19], and the coreset size is then reduced iteratively using Theorem 3.1, which simplifies the analysis greatly.

4.4 Graphs with Bounded Highway Dimension

The notion of highway dimension was proposed by Abraham, Fiat, Goldberg, and Werneck [AFGW10] to measure the complexity of road networks. Motivated by the empirical observation that a shortest path between two far-away cities always passes through a small number of hub cities, the highway dimension is defined, roughly speaking, as the maximum size of a hub set that meets every long shortest path, where the maximum is over all localities of all distance scale. Several slightly different definitions of highway dimension appear in the literature, and we use the one proposed in [FFKP18].

Definition 4.20 (Highway Dimension [FFKP18])**.** Fix some universal constant $\rho \geq 4$. The highway dimension of an edge-weighted graph $G = (V, E)$, denoted $\mathrm{hdim}(G)$, is the smallest integer t such that for every $r \geq 0$ and $x \in V$, there is a subset $S \subseteq B(x, \rho r)$ with $|S| \leq t$, such that S intersects every shortest path of length at least r all of whose vertices lie in $B(x, \rho r)$.

Remark 4.21. This version generalizes the original one from [AFGW10] (and also the subsequent journal version [ADF+16]), and it was shown to capture a broader range of real-world transportation networks [FFKP18]. We also note that the version in [ADF+16] is stronger than the notion of doubling dimension [GKL03], however, the version that we use (from [FFKP18]) is not. In particular, it means that the previous coreset result for doubling metrics [HJLW18] does not apply to our case.

Unlike the excluded-minor and Euclidean cases mentioned in earlier sections, our coresets for graphs with bounded highway dimension are obtained using terminal embeddings with an additive distortion.

[11] We assume that evaluating $\|x - y\|_2$ for $x, y \in \mathbb{R}^m$ takes time $O(m)$.

Lemma 4.22. *Let $G = (V, E)$ be an edge-weighted graph and denote its shortest-path metric by $M(V, d)$. Then for every $0 < \epsilon < 1/2$, weighted set $X \subseteq V$ and an (unweighted) subset $S \subseteq V$, there exists $\mathcal{F}_S = \{f_x : V \to \mathbb{R}_+ \mid x \in X\}$ such that*

$$\forall x \in X, c \in V, \quad d(x, c) \leq f_x(c) \leq (1 + \epsilon) \cdot d(x, c) + \epsilon \cdot d(x, S),$$

and $\mathrm{sdim}_{\max}(\mathcal{F}_S) = (|S| + \mathrm{hdim}(G))^{O(\log(1/\epsilon))}$.

Proof. We rely on an embedding of graphs with bounded highway dimension into graphs with bounded treewidth, as follows.

Lemma 4.23 ([BKS18])**.** *For every $0 < \epsilon < 1/2$, edge-weighted graph $G = (V, E)$ of highway dimension h, and $S \subseteq V$, there exists a graph $G' = (V', E')$ of treewidth $\mathrm{tw}(G') = (|S|+h)^{O(\log(1/\epsilon))}$, and a mapping $\phi : V \to V'$ such that*

$$\forall x, y \in V, \quad d_G(x, y) \leq d_{G'}(\phi(x), \phi(y)) \leq (1 + \epsilon) \cdot d_G(x, y) + \epsilon \cdot \min\{d(x, S), d(y, S)\}.$$

We now apply on G' (the graph produced by Lemma 4.23), the following result from [BBH+20, Lemma 3.5], which produces the function set $\mathcal{F}_S$ we need for our proof.

Lemma 4.24 ([BBH+20])**.** *Let $G = (V, E)$ be an edge-weighted graph, and denote its shortest-path metric by $M(V, d)$. Then for every weighted set $X \subseteq V$, the function set $\mathcal{F} = \{d(x, \cdot) \mid x \in X\}$ has $\mathrm{sdim}_{\max}(\mathcal{F}) = O(\mathrm{tw}(G))$, where $\mathrm{tw}(G)$ is the treewidth of G.*

Notice that we could also apply on G' our own Lemma 4.1, because bounded-treewidth graphs are also excluded-minor graphs, however Lemma 4.24 has better dependence on $\mathrm{tw}(G)$ and also saves a $\mathrm{poly}(1/\epsilon)$ factor. This concludes the proof of Lemma 4.22. □

Corollary 4.25 (Coresets for Graphs with Bounded Highway Dimension)**.** *For every edge-weighted graph $G = (V, E)$, $0 < \epsilon, \delta < 1/2$, and integer $k \geq 1$, k-Median of every weighted set $X \subseteq V$ (with respect to the shortest path metric of G) admits an ϵ-coreset of size $\tilde{O}((k+\mathrm{hdim}(G))^{O(\log(1/\epsilon))}) \log \frac{1}{\delta})$. Furthermore, it can be computed in time $\tilde{O}(|E|)$ with success probability $1 - \delta$.*

Proof. By combining Lemma 3.9, Corollary 3.8 with our terminal embedding from Lemma 4.22, we obtain an efficient also for constructing a coreset of the said size. Notice that we do not need to apply the iterative size reduction (Theorem 3.1) because $\mathrm{sdim}_{\max}$ is independent of X, thanks to the additive error. □

5 Applications: Improved Approximation Schemes for k-Median

In this section, we apply coresets to design approximation schemes for k-Median in shortest-path metrics of planar graphs and graphs with bounded highway dimension. In particular, we give an FPT-PTAS, parameterized by k and ϵ, for k-Median in graphs with bounded highway dimension, and a PTAS for k-Median in planar graphs. Both algorithms run in time near-linear in $|V|$ and improve state of the art results.

FPT-PTAS An ϵ-coreset D reduces the size of the input data set X while approximately preserving the cost for all clustering centers. Intuitively, in order to find a $(1+\epsilon)$-approximate solution, it suffices to solve the problem on D instead of X. However, solving the problem on D does not necessarily imply a PTAS for X because the optimal center C maybe contain element from the ambient space V, and thus would require enumerating all center sets from V^k making this approach prohibitively expensive. Instead, we enumerate all k-partitions of D and find an optimal center for each part. This simple idea implies an FPT-PTAS for k-Median and it can be implemented efficiently if the coreset size is independent of the input X. We formalize this idea in Section 5.1.

Centroid Set The aforementioned simple idea of enumerating all k-partitions of the coreset has exponential dependence in k, and hence is not useful for PTAS. Precisely, the bottleneck is that the set of potential centers, which is V, is not reduced. To reduce the potential center set, we consider *centroid set* that was first introduced by [Mat00] in the Euclidean setting, and later has been extended to other settings, e.g., doubling spaces [HJLW18]. A centroid set is a subset of V that contains a $(1+\epsilon)$-approximate solution. We obtain centroid sets of size *independent* of the input X for planar k-Median, improving the recent bound of $(\log |V|)^{\epsilon^{-O(1)}}$ from [CPP19]. The formal statement of our result for the centroid set can be found in Section 5.2.

PTAS for Planar k-Median The aforementioned improvement for centroid sets immediately implies improved PTAS for k-Median. Indeed, a $(1+\epsilon)$-approximation for the centroid set is as well a $(1+\Theta(\epsilon))$-approximation for the original data set. Specifically, we apply our centroid set to speedup a local search algorithm [CKM19] for planar k-Median, and our result is a PTAS that runs in time $\tilde{O}\left((k\epsilon^{-1})^{\epsilon^{-O(1)}}|V|\right)$ which is near-linear in $|V|$. This improves a previous PTAS [CKM19] whose running time is $k^{O(1)}|V|^{\epsilon^{-O(1)}}$, and an FPT-PTAS [CPP19] whose running time is $2^{O(k\epsilon^{-3}\log(k\epsilon^{-1}))}|V|^{O(1)}$. Details of the PTAS can be found in Section 5.3.

5.1 FPT-PTAS

We state our FPT-PTAS as a general reduction. Specifically we show that if a graph family admits a small ϵ-coreset then it also admits an efficient FPT-PTAS.

Lemma 5.1. *Let $\mathcal{G}$ be family of graphs. Suppose for $0 < \epsilon < \frac{1}{2}$, integer $k \geq 1$, every graph $G = (V, E) \in \mathcal{G}$ and every weighted set $X \subseteq V$, there is an ϵ-coreset $D = D(G, X)$ for k-Median on X in the shortest-path metric of G. Then there exists an algorithm that for every $0 < \epsilon < \frac{1}{2}$, integer $k \geq 1$ and $G \in \mathcal{G}$ computes a $(1+\epsilon)$-approximate solution for k-Median on any weighted set $X \subseteq V$ in time $\tilde{O}(k^{1+\|D(G,X)\|_0}|V|)$.*

Proof. The algorithm finds an *optimal* solution for the weighted instance defined by D. This optimal solution is a $(1+\epsilon)$-approximate solution for k-Median on the original data set X since D is an ϵ-coreset.

To find the optimal solution for k-Median on D we enumerate all k-clusterings (i.e. k-partitions) $\mathcal{C} = \{C_1, \ldots, C_k\}$ of D. For each part C_i we find an optimal center $c_i \in V$ that minimizes the cost of part C_i, i.e. $\min_{c_i \in V} \sum_{x \in C_i} w_D(x) \cdot d(x, c_i)$. The optimal solution is the k-center set that achieves the minimum total cost over all such k-clusterings of D.

To implement this algorithm efficiently, we first pre-compute all distances between point in D and points in V. This can be done in time $\tilde{O}(\|D\|_0|V|)$ using e.g. Dijkstra' algorithm. Using the pre-computed distances, we can find, in $O(|V|)$ time, the optimal center for any fixed set $C \subseteq D$.

Since there are k parts $C_1, \ldots, C_k$ and since there are $k^{\|D\|_0}$ possible partitions, the total running time is $\tilde{O}\left(k^{1+\|D\|_0}|V|\right)$. This completes the proof. □

FPT-PTAS for Graphs with Bounded Highway Dimension Combining Lemma 5.1 with Corollary 4.25, we obtain an FPT-PTAS for k-Median in graphs of bounded highway dimension. Compared with the previous bound $|V|^{O(1)} \cdot f(\epsilon, k, \mathrm{hdim}(G))$ from [BKS18, Theorem 2], our result runs in time near-linear in $|V|$ which is a significant improvement. Moreover, our algorithm is based on straightforward enumeration while [BKS18] is based on dynamic programming.

Corollary 5.2. *There is an algorithm that for every $0 < \epsilon < \frac{1}{2}$, integer $k \geq 1$, every edge-weighted graph $G = (V, E)$, computes a $(1+\epsilon)$-approximate solution for k-Median on every weighted set $X \subseteq V$ with constant probability, running in time $\tilde{O}\left(|V| \cdot k^{(k+\mathrm{hdim}(G))^{O(\log \epsilon^{-1})}}\right)$.*

Similarly, plugging Corollary 4.2 into Lemma 5.1 yields an FPT-PTAS for k-Median in planar graphs. We do not state this result here because the improved PTAS in the following section has a better running time.

5.2 Centroid Sets

The focus of the section is to present an improved centroid set that will be combined with a local search algorithm to yield a better PTAS. As already mentioned, a centroid set is a subset of points that contains a near-optimal solution. The formal definition is given below, and our centroid set is presented in Theorem 5.4.

Definition 5.3 (Centroid Set)**.** Given a metric space $M(V, d)$ and weighted set $X \subseteq V$, a set of points $S \subseteq V$ is an ϵ-centroid set for (k, z)-Clustering on X if there is a center set $C \in S^k$ such that $\mathrm{cost}_z(X, C) \leq (1+\epsilon) \cdot \mathrm{OPT}_z(X)$.

Theorem 5.4. *There is an algorithm that computes an ϵ-centroid set D of size*

$$\|D\|_0 = (\epsilon^{-1})^{O(\epsilon^{-2})} \mathrm{poly}(\|X\|_0),$$

for every $0 < \epsilon < \frac{1}{2}$, every planar graph $G = (V, E)$ and weighted subset $X \subseteq V$, running in time $\tilde{O}((\epsilon^{-1})^{O(\epsilon^{-2})} \mathrm{poly}(\|X\|_0)|V|)$.

First of all, we show there is a near-optimal solution $C^\star$ such that the distance from every center in $C^\star$ to X can only belong to $\mathrm{poly}(\|X\|_0)$ number of distinct distance scales. This is an essential property to achieve centroid sets of size independent of V. Specifically, consider the pairwise distance between points in X, and assume they are sorted as

$$d_1 \leq d_2 \leq \ldots \leq d_m$$

where $m = \binom{\|X\|_0}{2}$. We prove the following lemma.

Lemma 5.5. *For every $\epsilon \in (0, 1/2)$, there is a k-subset $C \subseteq V^k$ and an assignment $\pi : X \to C$, such that*

$$\sum_{x \in X} w_X(x) \cdot d(x, \pi(x)) \leq (1+2\epsilon) \cdot \mathrm{OPT}(X), \tag{21}$$

and for every $x \in X$, $d(x, \pi(x))$ belongs to an interval $\mathcal{I}_j := [\epsilon d_j, d_j/\epsilon]$ for some $j = 1, \ldots, m$. In particular, C is a $(1+2\epsilon)$-approximation to k-Median on X.

Proof. Let $C^\star = \{c_1^\star, \ldots, c_k^\star\} \subseteq V$ be the optimal solution to k-MEDIAN on X, and we will define C by "modifying" $C^\star$. Let $C_i^\star \subseteq X$ be the corresponding cluster of $c_i^\star$ and define $\mathrm{cost}_i^\star := \sum_{x \in C_i^\star} w_X(x) \cdot d(x, c_i^\star)$ to be the cost contributed by $C_i^\star$.

The proof strategy goes as follows. We examine $c_i^\star \in C^\star$ one by one. For each $c_i^\star$, we will define $c_i \in C$ as some point in $C_i^\star$, and the assignment π assigns every point in $C_i^\star$ to c_i. To bound the cost, we will prove $\sum_{x \in C_i^\star} w_X(x) \cdot d(x, c_i) \le (1 + 2\epsilon) \cdot \mathrm{cost}_i^\star$ for each i, and this implies (21).

Now fix some i. If $c_i^\star$ satisfies for every $x \in X$, there is some $1 \le j \le m$ such that $d(x, c_i^\star)$ belongs to $\mathcal{I}_j = [\epsilon d_j, \epsilon^{-1} d_j]$, then we include $c_i := c_i^\star$ to C, and for all $x \in C_i^\star$, let $\pi(x) := c_i^\star$. Since the center $c_i^\star$ is included in C as is, the cost corresponding to $C_i^\star$ is not changed.

Otherwise, there is some $\hat{x} \in X$ such that for every $1 \le j \le m$, either $d(\hat{x}, c_i^\star) < \epsilon d_j$ or $d(\hat{x}, c_i^\star) > \epsilon^{-1} d_j$. Then we pick any such $\hat{x}$, let $c_i := \hat{x}$, and define for each $x \in C_i^\star$, $\pi(x) := \hat{x}$. We note that for every $x' \in X$, $d(\hat{x}, x')$ equals some d_j by definition, so $d(\hat{x}, x') = d_j \in \mathcal{I}_j$.

Hence, it remains to prove that the cost is still bounded, i.e. $\sum_{x \in C_i^\star} w_X(x) \cdot d(x, \hat{x}) \le (1 + 2\epsilon) \cdot \mathrm{cost}_i^\star$, and we prove it by showing $\forall x \in C^\star$, $d(x, \hat{x}) \le (1 + 2\epsilon) \cdot d(x, c_i^\star)$. Observe that $d(x, \hat{x}) = d_j$ for some j, so depending on whether $d(\hat{x}, c_i^\star) < \epsilon d_j$ or $d(\hat{x}, c_i^\star) > \epsilon^{-1} d_j$ we have two cases.

- If $d(\hat{x}, c_i^\star) < \epsilon d_j = \epsilon d(x, \hat{x})$, then by triangle inequality, $d(x, \hat{x}) \le d(x, c_i^\star) + d(c_i^\star, \hat{x}) \le d(x, c_i^\star) + \epsilon d(x, \hat{x})$, hence $d(x, \hat{x}) \le \frac{1}{1-\epsilon} d(x, c_i^\star) \le (1 + 2\epsilon) d(x, c_i^\star)$ when $\epsilon \in (0, 1/2)$.
- Otherwise, $d(\hat{x}, c_i^\star) > \epsilon^{-1} d_j$, by triangle inequality, $d(x, c_i^\star) \ge d(\hat{x}, c_i^\star) - d(x, \hat{x}) > \frac{1-\epsilon}{\epsilon} d(x, \hat{x})$, which implies $d(x, \hat{x}) < \frac{\epsilon}{1-\epsilon} d(x, c_i^\star) < (1 + \epsilon) d(x, c_i^\star)$ when $\epsilon \in (0, 1/2)$.

This completes the proof. □

Proof of Theorem 5.4. Suppose $C^\star$ is an optimal solution. Our general proof strategy is to find a point c' that is sufficiently close to c for very center point $c \in C^\star$. Specifically, consider a center point $c \in C^\star$, and let $x_c \in X$ be the *closest* point to it. We want to guarantee that there always exists some c' in the centroid set, such that $d(c, c') \le \epsilon \cdot d(c, x_c)$, and this would imply the error guarantee of the centroid set by triangle inequality.

Since we can afford $(1 + \epsilon)$-multiplicative error, we round the distances to the nearest power of $(1 + \epsilon)$. Furthermore, we can assume without loss of generality that $C^\star$ is the $(1 + \epsilon)$-approximate solution claimed by Lemma 5.5, Then by Lemma 5.5, any distance $d(c, x)$ for $c \in C^\star$ and $x \in X$ has to lie in some interval $\mathcal{I}_j = [\epsilon d_j, \epsilon^{-1} d_j]$, and because of the rounding of distances, the distances on $C^\star \times X$ have to take from a set $\{r_1, \ldots, r_t\}$, where $t = \mathrm{poly}(\epsilon^{-1} \|X\|_0)$.

However, $C^\star$ is not known by the algorithm, and we have to "guess" c and c'. Specifically we enumerate over all points $x \in X$ which corresponds to the nearest point of c, and connection costs $r \in \{r_1, ..., r_t\}$ corresponding to $d(x, c)$, where c is some imaginary center in $C^\star$. To implement this efficiently, we pre-process the distances on $V \times X$ using $O(\|X\|_0)$ runs of Dijkstra's algorithm in time $\tilde{O}(\|X\|_0 |V|)$, and then r_i's are enumerated in $O(t)$ time.

Then to find c', a naive approach is to add an ϵr-net of $B(x, r)$ into D. The problem is that there may be too many points in the ϵ-net, so we need to use the structure of the graph to construct the net more carefully, and we make use of Lemma 4.5 which is restated as follows.

Lemma 5.6 (Restatement of Lemma 4.5)**.** *For every edge-weighted planar graph $G = (V, E)$ and subset $S \subseteq V$, V can be broken into parts $\Pi := \{V_i\}_i$ with $|\Pi| = \mathrm{poly}(|S|)$ and $\bigcup_i V_i = V$, such that for every $V_i \in \Pi$,*

1. *$|S \cap V_i| = O(1)$,*
2. *there exists a collection of shortest paths $\mathcal{P}_i$ in G with $|\mathcal{P}_i| = O(1)$ and removing the vertices of all paths in $\mathcal{P}_i$ disconnects V_i from $V \setminus V_i$ (points in V_i are possibly removed).*

Furthermore, such Π *and the corresponding shortest paths* $\mathcal{P}_i$ *for* $V_i \in \Pi$ *can be computed in* $\tilde{O}(|V|)$ *time.*

Apply Lemma 5.6 with (unweighted) $S = X$ to compute parts Π and the corresponding shortest paths $\mathcal{P}_i := \{P_j\}_j$ for each $V_i \in \Pi$, in $\tilde{O}(|V|)$ time. Then, apart from enumerating x_c and r, we further enumerate the set $V_i \in \Pi$. For each $P_j^i \in \mathcal{P}_i$, we let $Q_j^i := P_j^i \cap B(x_c, \epsilon^{-1}r + r)$. Observe that P_j^i is a path, so by triangle inequality Q_j^i is contained in a segment of length $O(\epsilon^{-1}r)$ of P_j^i. We further find an ϵr-net[12] R_j^i for Q_j^i which is of size $O(\epsilon^{-2})$. Finally, we let $R_i := \bigcup_j R_j^i$ denote the union of net points in all the shortest paths in $\mathcal{P}_i$, and $R_i' := R_i \cup (X \cap V_i)$ as the set with $X \cap V_i$ included in R_i. By Lemma 5.6, we know $|X \cap V_i| = O(1)$. Write $R_i' = \{y_1, \ldots, y_m\}$. We consider the set of possible distance tuples to R_i', i.e. for a point x, we consider the vector $(d(x, y_1), \ldots, d(x, y_m))$.

To restrict the number of possible distance tuples, we need to carefully discretize the distances so that the distances only come from a small ground set.

- For $y \in X \cap V_i$, because of Lemma 5.5, we can discretize and assume $d(x, y)$ from $\{r_1, ..., r_t\}$.
- For $y \in R_i$, we note that we will only use $d(x, y)$ such that $d(x, y) = O(r/\epsilon)$, so we only need to take $d(x, y)$ from $\{0, \epsilon r, 2\epsilon r, ..., (\epsilon^{-2} + 5)\epsilon r\}$ (noting that here we use an ϵr additive stepping).

Since $|R_i| = O(\epsilon^{-2})$ and $|X \cap V_i| = O(1)$, there are $(\epsilon^{-2})^{O(\epsilon^{-2})} t^{O(1)}$ many possible tuples.

For every tuple $(a_1, \ldots, a_m)$, we find an arbitrary point x' in $V_i \cap B(x_c, r)$ (if it exists) that realizes the distance tuple to R_i' when rounding to the closest discretized distance, i.e. $d'(x', y_i) = a_i$ for $1 \le i \le m$ where d' is the discretized distance, and add x' into D.

In total, we have added $(\epsilon^{-2})^{O(\epsilon^{-2})} t^{O(1)} \operatorname{poly}(\|X\|_0) = (\epsilon^{-1})^{O(\epsilon^{-2})} \operatorname{poly}(\|X\|_0)$ points into D, as desired. This whole process of enumerating V_i, computing ϵr-nets and finding point x' for each tuple can be implemented in time $O((\epsilon^{-1})^{O(\epsilon^{-2})} \operatorname{poly}(\|X\|_0)|V|)$.

Error Analysis We will prove D is indeed an ϵ-centroid set. Consider the solution $C^\star = \{c_1, \ldots, c_k\}$ and the corresponding assignment π guaranteed by Lemma 5.5. Suppose $C^\star$ clusters X into $\{C_1^\star, \ldots, C_k^\star\}$ by the arrangement π. We will prove the following claim.

Claim 5.7. *For every* $1 \le i \le k$*, there exists* $c_i' \in D$ *such that*

$$\forall y \in C_i^\star, \quad d(y, c_i') \le (1 + O(\epsilon)) \cdot d(y, c_i), \tag{22}$$

where $C_i^\star \subseteq X$ *is the cluster of* X *corresponding to* $c_i \in C^\star$*.*

Suppose the above claim is true, then we define a k-subset $C' := \{c_1', \ldots, c_k'\}$, and it implies that $\operatorname{cost}(X, C') \le (1 + O(\epsilon)) \cdot \operatorname{cost}(X, C^\star)$. Hence, it remains to prove Claim 5.7,

Proof of Claim 5.7. Fix $1 \le i \le k$. We start with defining c_i'. Suppose $x_{c_i} \in X$ is the closest point to c_i and let $r_i := d(x_{c_i}, c_i)$. Let $V_j \in \Pi$ such that $c_i \in V_j$, and consider the moment that our algorithm enumerates x_{c_i}, r_i and V_j. By construction, we have the following fact.

Fact 5.8. *There exists some point* $c' \in D$ *such that*

1. $d(x_{c_i}, c') \le r_i$

[12] For $\rho > 0$ and some subset $W \subseteq V$, a ρ-net is a subset $Y \subseteq V$ such that $\forall x, y \in Y$, $d(x, y) \ge \rho$ and $\forall x \in W$ there is $y \in Y$ with $d(x, y) < \rho$.

2. *for every* $y \in R_i$, *if* $d(c_i, y) \le (\epsilon^{-2}+4)\epsilon r_i$, *then* $d(c', y) \in d(c_i, y) \pm \epsilon r_i$

3. *for every* $y \in X \cap V_i$, $d(c', y) \in (1 \pm \epsilon) \cdot d(c_i, y)$.

We pick c_i' as any of such c' in Fact 5.8.

Now we analyze the error. Fix $y \in C_i^\star$. We note that the R_i' that we pick only covers an $O(\epsilon^{-1} r_i)$ range, so even though c_i' approximate c_i on the distance tuples, it cannot directly imply the distance from c_i' to all other points in $C_i^\star$ is close to that from c_i, and we need the following argument.

- If $d(y, c_i) > \epsilon^{-1} r_i$, then y is far away and $d(y, c_i')$ cannot be handled by the distance tuples. However, we observe that in this case $d(c_i, c_i')$ is small relative to $d(y, c_i)$. In particular, we have $d(c_i, c_i') \le d(c_i, x_{c_i}) + d(c_i', x_{c_i}) \le 2r_i$. Hence, it implies

$$d(y, c_i') \le d(y, c_i) + d(c_i, c_i') \le d(y, c_i) + 2r_i \le (1+2\epsilon) \cdot d(y, c_i).$$

- Otherwise, $d(y, c_i) \le \epsilon^{-1} r_i$, and we will use that c_i' and c_i are close with respect to the tuple distance, and use the separating shortest paths $\mathcal{P}_j$ (recalling that $V_j \in \Pi$ is the part that c_i belongs to).

  - If $y \in V_j$, then y belongs to the set R_j' and $d(c_i', y)$ belongs to one of the distance tuples (recalling that $y \in C_i^\star \subseteq X$). Hence, by the guarantee of the distance tuples, c_i' satisfies $d(y, c_i') = d(y, c_i)$.
  - Otherwise, $y \notin V_j$. Then the shortest path $y \rightsquigarrow c_i$ has to pass through at least one of the shortest paths in $\mathcal{P}_j$. Now suppose $P_l^j \in \mathcal{P}_j$ is the separating shortest path that shortest path $y \rightsquigarrow c_i$ passes through. Since $d(y, c_i) \le \epsilon^{-1} r_i$, we have $y \in B(x_{c_i}, \epsilon^{-1} r_i + r_i)$. Since P_l^j is a shortest path in G, $y \rightsquigarrow c_i$ can only cross it once.

    Hence, there is $y', y'' \in Q_l^j$ such that

$$d(y, c_i) = d(y, y') + d(y', y'') + d(y'', c_i).$$

    Since R_l^j is an ϵr_i-net of Q_l^j, by triangle inequality, we know there exists $z', z'' \in R_l^j$ such that

$$d(y, z') + d(z', z'') + d(z'', c_i) \le d(y, c_i) + 4\epsilon r_i.$$

    Since $d(z'', c_i) \le d(y, c_i) + 4\epsilon r_i \le (\epsilon^{-2}+4)\epsilon r_i$, by Fact 5.8 we know that $d(z'', c_i') \le d(z'', c_i) + \epsilon r_i$. Finally, by triangle inequality, we have,

$$d(y, c_i') \le d(y, z') + d(z', z'') + d(z'', c_i') \le d(y, c_i) + O(\epsilon r_i),$$

    Observe that by definition $d(y, c_i) \ge d(x_{x_i}, c_i) = r_i$, so we conclude that

$$d(y, c_i') \le d(y, c_i) + O(\epsilon r_i) \le (1 + O(\epsilon)) \cdot d(y, c_i).$$

This completes the proof of Claim 5.7. □

This completes the proof of Theorem 5.4 □

5.3 Improved PTAS's for Planar k-Median

Recently, [CKM19] showed the local search algorithm that swaps $\epsilon^{-O(1)}$ points in the center set in each iteration yields a $1+\epsilon$ approximation for k-MEDIAN in planar and the more general excluded-minor graphs. We use the centroid set and coreset to speedup this algorithm, and we obtain the following PTAS.

Corollary 5.9. *There is an algorithm that for every $0 < \epsilon < \frac{1}{2}$, integer $k \geq 1$ and every edge-weighted planar graph $G = (V, E)$, computes a $(1+\epsilon)$-approximate solution for k-MEDIAN on every weighted set $X \subseteq V$ with constant probability, running in time $\tilde{O}((\epsilon^{-1}k)^{\epsilon^{-O(1)}} \cdot |V|)$.*

As noted by [HJLW18] and [FRS19], the potential centers that the local search algorithm should consider can be reduced using an ϵ-centroid set, but to make the local search terminate properly, we also need to evaluate the objective value accurately in each iteration , which means we also need a coreset. Hence, we start with constructing a coreset using Corollary 4.2, and then extend it to be a centroid set using Theorem 5.4.

Proof of Corollary 5.9. Construct an ϵ-coreset S of size $\text{poly}(\epsilon^{-1}k)$ using Corollary 4.2, and apply Theorem 5.4 with $X = S$ to obtain an ϵ-centroid set S' of size $(\epsilon^{-1})^{O(\epsilon^{-2})}k^{O(1)}$. Then the algorithm constructs a weighted set D that consists of $S \cup S'$, and the weights of points $x \in S$ is set to $w_S(x)$, and those $x \in S' \setminus S$ has weight 0. It is immediate that D is both an ϵ-coreset and an ϵ-centroid set, whose size is $\|D\|_0 = (\epsilon^{-1})^{O(\epsilon^{-2})}k^{O(1)}$. We pre-process the pairwise distance in $D \times D$ using Dijkstra's algorithm. The overall running time for all these steps is $\tilde{O}((\epsilon^{-1})^{\epsilon^{-O(1)}}k^{O(1)} \cdot |V|)$.

We next use D to accelerate [CKM19, Algorithm 1]. The algorithm first defines an initial center set C to be an arbitrary subset of D. Then in each iteration, the algorithm enumerates $C' \in D^k$ that is formed by swapping at most $\epsilon^{-O(1)}$ points in D from C. Update $C := C'$ if some C' has cost $\text{cost}(D, C') \leq (1 - \frac{\epsilon}{|V|}) \cdot \text{cost}(D, C)$, and terminate otherwise. The running time for each iteration is $(\epsilon^{-1}k)^{\epsilon^{-O(1)}}$.

By [CKM19], the algorithm always finds a $(1+\epsilon)$-solution when it terminates, and the number of iterations is at most $\epsilon^{-1}|V|$ until termination. Therefore, the total running time is bounded by $\tilde{O}((\epsilon^{-1}k)^{\epsilon^{-O(1)}} \cdot |V|)$. This completes the proof. □

References

[ADF+16] Ittai Abraham, Daniel Delling, Amos Fiat, Andrew V. Goldberg, and Renato F. Werneck. Highway dimension and provably efficient shortest path algorithms. *J. ACM*, 63(5):41:1–41:26, 2016.

[AFGW10] Ittai Abraham, Amos Fiat, Andrew V. Goldberg, and Renato Fonseca F. Werneck. Highway dimension, shortest paths, and provably efficient algorithms. In *SODA*, pages 782–793. SIAM, 2010.

[AG06] Ittai Abraham and Cyril Gavoille. Object location using path separators. In *PODC*, pages 188–197. ACM, 2006. doi:10.1145/1146381.1146411.

[ARR98] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for Euclidean k-medians and related problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 106–113, 1998.

[BBCA+19] Luca Becchetti, Marc Bury, Vincent Cohen-Addad, Fabrizio Grandoni, and Chris Schwiegelshohn. Oblivious dimension reduction for k-means: beyond subspaces and the johnson-lindenstrauss lemma. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1039–1050, 2019.

[BBH+20] Danial Baker, Vladimir Braverman, Lingxiao Huang, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering in graphs of bounded treewidth. In *ICML*, Proceedings of Machine Learning Research, 2020. To appear. arXiv:1907.04733.

[BFL16] Vladimir Braverman, Dan Feldman, and Harry Lang. New frameworks for offline and streaming coreset constructions. *CoRR*, abs/1612.00889, 2016. arXiv:1612.00889.

[BJKW19] Vladimir Braverman, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for ordered weighted clustering. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 744–753. PMLR, 2019.

[BKS18] Amariah Becker, Philip N. Klein, and David Saulpic. Polynomial-time approximation schemes for k-center, k-median, and capacitated vehicle routing in bounded highway dimension. In *ESA*, volume 112 of *LIPIcs*, pages 8:1–8:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. https://arxiv.org/abs/1707.08270. doi:10.4230/LIPIcs.ESA.2018.8.

[BPR+14] Jarosław Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k-median, and positive correlation in budgeted optimization. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 737–756. SIAM, 2014.

[BT15] Nicolas Bousquet and Stéphan Thomassé. VC-dimension and Erdős–Pósa property. *Discret. Math.*, 338(12):2302–2317, 2015.

[CEM+15] Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 163–172, 2015.

[Che09] Ke Chen. On coresets for k-Median and k-Means clustering in metric and Euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009. doi:10.1137/070699007.

[CKM19] 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.

[CPP19] Vincent Cohen-Addad, Marcin Pilipczuk, and Michal Pilipczuk. Efficient approximation schemes for uniform-cost clustering problems in planar graphs. In *ESA*, volume 144 of *LIPIcs*, pages 33:1–33:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[CW15] Kenneth L. Clarkson and David P. Woodruff. Sketching for M-estimators: A unified approach to robust regression. In *SODA*, pages 921–939. SIAM, 2015.

[CZQ+08] Weiwei Cui, Hong Zhou, Huamin Qu, Pak Chung Wong, and Xiaoming Li. Geometry-based edge clustering for graph visualization. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1277–1284, 2008.

[EKM14] David Eisenstat, Philip N. Klein, and Claire Mathieu. Approximating k-center in planar graphs. In *SODA*, pages 617–627. SIAM, 2014. doi:10.1137/1.9781611973402.47.

[FFKP18] Andreas Emil Feldmann, Wai Shing Fung, Jochen Knemann, and Ian Post. A $(1+\varepsilon)$-embedding of low highway dimension graphs into bounded treewidth graphs. *SIAM Journal on Computing*, 47(4):1667–1704, 2018. doi:10.1137/16M1067196.

[FFS06] Dan Feldman, Amos Fiat, and Micha Sharir. Coresets for weighted facilities and their applications. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, FOCS 06, page 315324. IEEE Computer Society, 2006. doi:10.1109/FOCS.2006.22.

[FKW19] Zhili Feng, Praneeth Kacham, and David P. Woodruff. Strong coresets for subspace approximation and k-median in nearly linear time. *CoRR*, abs/1912.12003, 2019. URL: http://arxiv.org/abs/1912.12003, arXiv:1912.12003.

[FL11] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *STOC*, pages 569–578. ACM, 2011. https://arxiv.org/abs/1106.1379.

[FMSW10] Dan Feldman, Morteza Monemizadeh, Christian Sohler, and David P. Woodruff. Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 10, page 630649. SIAM, 2010.

[FRS19] Zachary Friggstad, Mohsen Rezapour, and Mohammad R. Salavatipour. Local search yields a PTAS for k-means in doubling metrics. *SIAM J. Comput.*, 48(2):452–480, 2019.

[FSS20] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM Journal on Computing*, 49(3):601–657, 2020. doi:10.1137/18M1209854.

[GI03] Venkatesan Guruswami and Piotr Indyk. Embeddings and non-approximability of geometric problems. In *SODA*, volume 3, pages 537–538, 2003.

[GKL03] Anupam Gupta, Robert Krauthgamer, and James R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *FOCS*, pages 534–543. IEEE Computer Society, 2003.

[Har11] Sariel Har-Peled. *On Complexity, Sampling, and ϵ-Nets and ϵ-Samples*, volume 173. American Mathematical Soc., 2011.

[HJLW18] Lingxiao Huang, Shaofeng H.-C. Jiang, Jian Li, and Xuan Wu. Epsilon-coresets for clustering (with outliers) in doubling metrics. In *FOCS*, pages 814–825. IEEE Computer Society, 2018.

[HJV19] Lingxiao Huang, Shaofeng H.-C. Jiang, and Nisheeth K. Vishnoi. Coresets for clustering with fairness constraints. In *NeurIPS*, pages 7587–7598, 2019.

[HK07] Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. *Discret. Comput. Geom.*, 37(1):3–19, 2007.

[HM04] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, STOC 04, page 291300. ACM, 2004. doi:10.1145/1007352.1007400.

[HV20] Lingxiao Huang and Nisheeth K. Vishnoi. Coresets for clustering in Euclidean spaces: importance sampling is nearly optimal. In *STOC*, pages 1416–1429. ACM, 2020.

[JJJ+00] Sugih Jamin, Cheng Jin, Yixin Jin, Danny Raz, Yuval Shavitt, and Lixia Zhang. On the placement of internet instrumentation. In *INFOCOM*, pages 295–304. IEEE Computer Society, 2000.

[JL84] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Conn., 1982)*, pages 189–206. Amer. Math. Soc., 1984. doi:10.1090/conm/026/737400.

[JMS02] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 731–740, 2002.

[JV01] Kamal Jain and Vijay V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *J. ACM*, 48(2):274–296, 2001.

[KL19] Zohar S. Karnin and Edo Liberty. Discrepancy, coresets, and sketches in machine learning. In *COLT*, volume 99 of *Proceedings of Machine Learning Research*, pages 1975–1993. PMLR, 2019.

[KM12] Philip Klein and Shay Mozes. Optimization algorithms for planar graphs. Book draft, http://www.planarity.org, 2012.

[LBK13] Yingyu Liang, Maria-Florina Balcan, and Vandana Kanchanapally. Distributed PCA and k-means clustering. In *The Big Learning Workshop at NIPS*, volume 2013. Citeseer, 2013.

[LFKF17] Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. Training gaussian mixture models at scale via coresets. *The Journal of Machine Learning Research*, 18(1):5885–5909, 2017.

[LGI+99] Bo Li, Mordecai J. Golin, Giuseppe F. Italiano, Xin Deng, and Kazem Sohraby. On the optimal placement of web proxies in the internet. In *INFOCOM*, pages 1282–1290. IEEE Computer Society, 1999.

[LMP13] Mu Li, Gary L. Miller, and Richard Peng. Iterative row sampling. In *FOCS*, pages 127–136. IEEE Computer Society, 2013.

[LS10] Michael Langberg and Leonard J. Schulman. Universal epsilon-approximators for integrals. In *SODA*, pages 598–607. SIAM, 2010.

[Mat00] Jivr'i Matouvsek. On approximate geometric k-clustering. *Discrete & Computational Geometry*, 24(1):61–84, 2000.

[MJF19] Alaa Maalouf, Ibrahim Jubran, and Dan Feldman. Fast and accurate least-mean-squares solvers. In *Advances in Neural Information Processing Systems*, pages 8305–8316, 2019.

[MMR19] Konstantin Makarychev, Yury Makarychev, and Ilya Razenshteyn. Performance of johnson-lindenstrauss transform for k-means and k-medians clustering. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1027–1038, 2019.

[MP04] Ramgopal R. Mettu and C. Greg Plaxton. Optimal time bounds for approximate clustering. *Mach. Learn.*, 56(1-3):35–60, 2004.

[MSSW18] Alexander Munteanu, Chris Schwiegelshohn, Christian Sohler, and David P. Woodruff. On coresets for logistic regression. In *NeurIPS*, pages 6562–6571, 2018.

[NN19] Shyam Narayanan and Jelani Nelson. Optimal terminal dimensionality reduction in Euclidean space. In *STOC*, pages 1064–1069. ACM, 2019. doi:10.1145/3313276.3316307.

[PT19] Jeff M Phillips and Wai Ming Tai. Near-optimal coresets of kernel density estimates. *Discrete & Computational Geometry*, pages 1–21, 2019.

[RMJ07] Matthew J. Rattigan, Marc E. Maier, and David D. Jensen. Graph clustering with network structure indices. In *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pages 783–790. ACM, 2007.

[SL97] Shashi Shekhar and Duen-Ren Liu. CCAM: A connectivity-clustered access method for networks and network computations. *IEEE Trans. Knowl. Data Eng.*, 9(1):102–119, 1997.

[SSS19] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means. In *WAOA*, volume 11926 of *Lecture Notes in Computer Science*, pages 232–251. Springer, 2019.

[SW18] Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. In *FOCS*, pages 802–813. IEEE Computer Society, 2018.

[TFL83] Barbaros C Tansel, Richard L Francis, and Timothy J Lowe. State of the artlocation on networks: a survey, part i and ii. *Management Science*, 29(4):482–497, 1983.

[Tho04] Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, 2004. doi:10.1145/1039488.1039493.

[Tho05] Mikkel Thorup. Quick k-Median, k-Center, and facility location for sparse graphs. *SIAM J. Comput.*, 34(2):405432, 2005. doi:10.1137/S0097539701388884.

[VC71] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971. doi:10.1137/1116025.

[VX12] Kasturi R. Varadarajan and Xin Xiao. On the sensitivity of shape fitting problems. In *FSTTCS*, volume 18 of *LIPIcs*, pages 486–497. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012. doi:10.4230/LIPIcs.FSTTCS.2012.486.

[YM04] Man Lung Yiu and Nikos Mamoulis. Clustering objects on a spatial network. In *SIGMOD Conference*, pages 443–454. ACM, 2004.

Appendices

A Proof of Lemma 4.5

Lemma A.1 (restatement of Lemma 4.5)**.** *For every edge-weighted planar graph $G = (V, E)$ and subset $S \subseteq V$, V can be broken into parts $\Pi := \{V_i\}_i$ with $|\Pi| = \text{poly}(|S|)$ and $\bigcup_i V_i = V$, such that for every $V_i \in \Pi$,*

1. *$|S \cap V_i| = O(1)$,*
2. *there exists a collection of shortest paths $\mathcal{P}_i$ in G with $|\mathcal{P}_i| = O(1)$ and removing the vertices of all paths in $\mathcal{P}_i$ disconnects V_i from $V \setminus V_i$ (points in V_i are possibly removed).*

Furthermore, such Π and the corresponding shortest paths $\mathcal{P}_i$ for $V_i \in \Pi$ can be computed in $\tilde{O}(|V|)$ time.

The proof of Lemma 4.5 is based on the following property of general trees. We note that the special case when $R = T$ was proved in [EKM14, Lemma 3.1] and our proof is based on it. Nonetheless, we provide the proof for completeness.

Lemma A.2. *Let T be a tree of degree at most 3 and let R be a subset of nodes in T. There is a partition of the nodes of T with $\text{poly}(|R|)$ parts, such that each part is a subtree of T that contains $O(1)$ nodes of R and has at most 4 boundary edges[13] connecting to the rest of T. Such partition can be computed in time $\tilde{O}(|T|)$, where $|T|$ is the number of nodes in T.*

Proof. We give an algorithm to recursively partition T in a top-down manner. The recursive algorithm takes a subtree T' as input, and if $|T' \cap R| \geq 4$, it chooses an edge e from T' and run recursively on the two subtrees T'_1 and T'_2 that are formed by removing e from T'. Otherwise, the algorithm simply declares the subtree T' a desired part and terminate, if $|T' \cap R| < 4$. Next, we describe how e is picked provided that $|T' \cap R| \geq 4$.

If T' has at most 3 boundary edges, we pick an edge $e \in T'$ such that each of the two subtrees T'_1, T'_2 formed by removing e satisfies $\frac{1}{3}|T' \cap R| \leq |T'_j \cap R| \leq \frac{2}{3}|T' \cap R|$, for $j = 1, 2$. By a standard application of the balanced separator theorem (see e.g. Lemma 1.3.1 of [KM12]), such edge always exists and can be found in time $O(|T'|)$.

Now, suppose T' has exactly 4 boundary edges. Then we choose an edge $e \in T'$, such that each of the two subtrees T'_1 and T'_2 formed by removing e has at most 3 boundary edges. Such e must exist because the maximum degree is at most 3, and such e may be found in time $O(|T'|)$ as well. To see this, suppose the four endpoints (in T') of the four boundary edges are a, b, c, d. It is possible that they are not distinct, but they can have a multiplicity of at most 2 because otherwise the degree bound 3 is violated. If any point has a multiplicity 2, say a and b, then it has to be

[13] Here a boundary edge is an edge that has exactly one endpoint in the subtree.

a leaf node in T' (again, because of the degree constraint), and we can pick the unique tree edge in T' connecting a as our e. Now we assume the four points are distinct, and consider the unique paths P_1, P_2 that connect a, b and c, d respectively. If P_1 and P_2 intersect, then the intersection must contain an edge as otherwise the intersections are at nodes only which means each of them have degree at least 4, a contradiction. Hence, we pick the intersecting edge as our e. Finally, if P_1 and P_2 are disjoint, we consider the unique path P_3 that connects a and c, and we pick edge $e := e'$ in P_3 that is outside both P_1 and P_2 to separate a and b from c and d.

We note that there are no further cases regarding the number of boundary edges of T', since in the case of 4 boundaries edges, both T'_1 and T'_2 have at most 3 boundary edges and it reduces to the first case.

It remains to analyze the size of the partition. By the property of balanced separator, we know that such recursive partition has $O(\log |R|)$ depth. Hence the total number of subtrees is $2^{O(\log |R|)} = \text{poly}(|R|)$. Finally, we note that in each level of depth, we scan the whole tree once, so the running time is upper bounded By $O(\log |R|) \cdot |T| = \tilde{O}(|T|)$. □

Proof of Lemma 4.5. We assume G is triangulated, since otherwise we can triangulate G and assign weight $+\infty$ to the new edges so that the shortest paths are the same as before. Let T be a shortest path tree of G from an arbitrary root vertex. Let $G^\star$ be the planar dual of G. Let $T^\star$ be the set of edges e of $G^\star$ such that the corresponding edge of e in G is not in T. Indeed, T and $T^\star$ are sometimes called *interdigitating* trees, and it is well known that $T^\star$ is a spanning tree of $G^\star$ (see e.g. [KM12]).

Choose $R^\star$ to be the set of faces that contain at least one point from S. We apply Lemma A.2 on $R = R^\star$ and $T = T^\star$ to obtain $\Pi^\star$, the collection of resulted subtrees of $T^\star$. Then $|\Pi^\star| = \text{poly}(|S|)$, and each part $C^\star$ in $\Pi^\star$ is a subset of faces in G such that only $O(1)$ of these faces contain some point in S on their boundaries. For a part $C^\star$ in $\Pi^\star$, let $V(C^\star)$ be the set of vertices in G that are contained in the faces in $C^\star$. Recall that G is triangulated, so each face can only contain $O(1)$ vertices from S on its boundary. Therefore, for each part $C^\star$ in $\Pi^\star$, $|C^\star \cap S| = O(1)$.

Still by Lemma A.2, each part $C^\star$ in $\Pi^\star$ corresponds to a subtree in $T^\star$, and it has at most 4 boundary edges connecting to the rest of $T^\star$. By the well-known property of planar duality (see e.g. [KM12]), each $C^\star$ is bounded by the fundamental cycles in T of the boundary edges. We observe that the vertices of a fundamental cycle lie on 2 shortest paths in G via the least common ancestor in T (recalling that T is the shortest path tree). So by removing at most 8 shortest paths in G, $V(C^\star)$ is disconnected from $V \setminus V(C^\star)$ for every $C^\star \in \Pi^\star$.

Therefore, we can choose $\Pi := \{V(C^\star) : C^\star \in \Pi^\star\}$. For the running time, we note that both the triangulation and the algorithm in Lemma A.2 run in $\tilde{O}(|V|)$ time. This completes the proof. □