Clustering in Varying Metrics

Deeparnab Chakrabarty* Jonathan Conroy* Ankita Sarkar*

Abstract

We introduce the *aggregated clustering* problem, where one is given T instances of a center-based clustering task over the same n points, but under different metrics. The goal is to open k centers to minimize an aggregate of the clustering costs—e.g., the average or maximum—where the cost is measured via k-center/median/means objectives. More generally, we minimize a norm Ψ over the T cost values.

We show that for $T \geq 3$, the problem is inapproximable to any finite factor in polynomial time. For $T = 2$, we give constant-factor approximations. We also show W[2]-hardness when parameterized by k, but obtain $f(k, T)\text{poly}(n)$-time 3-approximations when parameterized by both k and T.

When the metrics have structure, we obtain efficient parameterized approximation schemes (EPAS). If all T metrics have bounded ε-scatter dimension, we achieve a $(1+\varepsilon)$-approximation in $f(k, T, \varepsilon)\text{poly}(n)$ time. If the metrics are induced by edge weights on a common graph G of bounded treewidth tw, and Ψ is the sum function, we get an EPAS in $f(T, \varepsilon, \mathsf{tw})\text{poly}(n, k)$ time. Conversely, unless (randomized) ETH is false, any finite factor approximation is impossible if parametrized by only T, even when the treewidth is $\mathsf{tw} = \Omega(\text{poly} \log n)$.

1 Introduction

Clustering problems such as the k-supplier or the k-median problem are classic discrete optimization problems which have formed the bedrock of many problems in operations research. In its simplest form, the objective is to "open" k locations in a metric space such that some function (eg, max or sum) of the distances of clients to the nearest open facility is minimized. Such problems have been extensively studied over the past 50 years [HS86, Ple87, CKMN01, JV01, CGTS02, AGK+04, CL12, BPR+14, LS16, Swa16, CN19, KLS18, CGK20, MV20, MV21, CNS22, CMV22, DLR22, AISX24, CCS24], and we have a very good understanding on the approximability of many of these problems.

In this paper, we consider clustering problems *when the distance function underlying the metric space can change over time*. To give a toy example, imagine that the clients are households in a New England town, and the distances are travel times over the road network in the town. On one hand, travel times may be quite different in January and June given the weather differences over the months. On the other hand, one can't hope to open different locations in different seasons. Our decision ought to take into account the different costs over time, for example by selecting a set of facilities that minimizes the average cost over the year. Although this sounds a natural question, to our knowledge what happens to the complexity of clustering with changing metrics hasn't been studied so far.[1] Formally, we study the following problem.

Definition 1 (Aggregate Clustering Problems)**.** *We are given $V = F \cup C$, a finite set of n points where C is the set of clients and F is a set of facilities*[2]*. We are given T distance functions $d_t(\cdot, \cdot)$ over $V \times V$ which are symmetric and satisfy triangle-inequality. We are given a parameter $k \in \mathbb{Z}$. Finally, we are given an* aggregator *function $\Psi : \mathbb{R}^T_{\geq 0} \to \mathbb{R}_{\geq 0}$. The goal is to 'open' a subset $S \subseteq F$ of k facilities that minimize a certain objective function (depending on the problem variant). The objective functions are:*

*Department of Computer Science, Dartmouth College.
{deeparnab, jonathan.conroy.gr, ankita.sarkar.gr}@dartmouth.edu

[1] One may be tempted to simply perform clustering on a single metric space defined by average distances over all months. However, this does not work: the cost of a client at each timestep is computed by take a minimum over a set of facilities, and this minimum is not preserved when we take an average.

[2] F and C need not be disjoint, and one can imagine the case of $F = C$ as a special but instructive case.

- $\Psi\left(\text{cost}_\infty(d_t; S) : t \in [T]\right)$, *where* $\text{cost}_\infty(d_t; S) := \max_{j \in C} d_t(j, S)$, *in the* Ψ*-aggregate* k*-supplier problem.*[3]
- $\Psi\left(\text{cost}_1(d_t; S) : t \in [T]\right)$, *where* $\text{cost}_1(d_t; S) := \sum_{j \in C} d_t(j, S)$, *in the* Ψ*-aggregate* k*-median problem.*
- $\Psi\left(\text{cost}_z(d_t; S) : t \in [T]\right)$, *where* $\text{cost}_z(d_t; S) := \left(\sum_{j \in C} d_t(j, S)^z\right)^{1/z}$, *in the* Ψ*-aggregate* (k, z)*-clustering problem.*

When it is clear from context, we denote the objective function as $\text{aggcost}(\cdot)$.

The above problem also can be thought of as a stochastic/robust version of clustering under uncertainty where the uncertainty is over the distances. Note that when Ψ is the sum-function, we wish to find a solution which minimizes the average k-supplier/k-median cost of the returned solution S, where the average is over the T "scenarios" (we are assuming the full-information setting); when Ψ is the max-function, we get the robust optimization setting. In the literature [AGGN10, MV21, CMV22], stochastic/robust clustering has been studied when the metric is fixed but the uncertainty is over the client sets. We can consider a generalized version of aggregate clustering, which models client demands changing over time.

Definition 2 (Generalized Aggregate Clustering)**.** *Apart from the* t *distances, we have* t *weight functions* $w_t : C \to \mathbb{R}$. *In the objective functions, we modify* cost*s by multiplying* $d_t(j, S)$*'s with* $w_t(j)$. *More precisely,* $\text{cost}_z(d_t, S) = \left(\sum_{j \in C} w_t(j) \cdot d_t(j, S)^z\right)^{1/z}$ *and* $\text{cost}_\infty(d_t; S) = \max_{j \in C} w_t(j) d_t(j, S)$. *In particular, if* $w_t : C \to \{0, 1\}$, *then this models having different client sets at different times.*

All our upper bounds apply for generalized aggregate clustering, and all our lower bounds hold even for the vanilla version of aggregate clustering. In fact, in Observation 40 we present a weak equivalence between the two problems, by showing that any algorithm for vanilla aggregate clustering can solve generalized aggregate clustering with $\{0, 1\}$ weights, i.e. aggregate clustering where the client set is allowed to change over time.

1.1 Our Findings

General Metrics. In its full generality, unfortunately, the aggregrate clustering problem becomes very hard when the number of scenarios T becomes 3 or larger. In Theorem 3 we show that it is NP-hard to distinguish between zero and non-zero solutions when $T \geq 3$. To complement this result, we show (in Theorems 12 and 21) that when $T = 2$, both the (generalized) Ψ-aggregate k-median and k-supplier problems have polynomial time $O(1)$-approximations when Ψ is any norm. In fact, for k-supplier, we match the optimal 3-approximation that is known at $T = 1$ [HS86, Ple87]. Both the above results, at a very high level, follow from the fact that bipartite matching (or rather, partition matroid intersection) is tractable, while 3D-matching is NP-hard.

Given the above hardness for $T \geq 3$, we attempt to bypass it in two ways: (i) investigating fixed-parameter tractable approximation algorithms, and (ii) imposing additional structure over the metrics in various scenarios.

Constant-factor FPT approximation algorithms. The hardness for $T = 3$ clearly shows one can't expect $f(T) \cdot \text{poly}(n, k)$-time approximation algorithms. Moreover, we show (in Theorem 5) that it is also hard to obtain $f(k) \cdot \text{poly}(n, T)$-time approximation algorithms: we encode the k-hitting set problem as an instance of aggregate clustering, and our lower bound follows from the $W[2]$ hardness of k-hitting set [CFK$^+$15, Theorem 13.12]. On the other hand, if we allow k and T to be both fixed parameters, then (in Theorem 30) we obtain an $f(k, T) \cdot \text{poly}(n)$-time $(3 + \varepsilon)$-approximation. The high level idea for this is as follows: Suppose we have T partitions of a universe U, and we must decide if there is a k-sized subset that is a *simultaneous* hitting set for all T partitions. In $(k + 1)^{kT} \cdot \text{poly}(n)$-time, we can guess for each $t \in [T]$ and $i \in [k]$ the part in the t^{th} partition that the i^{th} hitting element hits. Such an idea is the core of many FPT approximation algorithms for clustering [ABB$^+$23, BGI25, GI25].

[3] We use the notation $d_t(j, S)$ to mean $\min_{i \in S} d_t(j, i)$.

Approximation schemes in structured metrics. To impose structure on the metric, we note that any finite metric d_t is a shortest path metric on an undirected *base graph* $G = (V, E)$ with weights $w_t(e)$ on the edges. Note that our restriction of having the *same* graph G across scenarios is without loss of generality if we allow a complete graph. Furthermore, in many applications, such as the road-network application mentioned in the early paragraphs of the introduction, this indeed is the case where the "weights" change over the seasons but the underlying network remains the same. Can one obtain better algorithmic results exploiting the structure of the underlying undirected base graph G?

Unfortunately, our hardness results apply even for very simple base graphs G. In particular, our reduction from hitting set (Theorem 5, which rules out any finite factor approximation algorithms in $f(k) \cdot \text{poly}(n, T)$ time) can be carried out even when G is a star. Moreover, our reduction from 3D matching (Theorem 3, which rules out any $f(T) \cdot \text{poly}(n, k)$-time finite factor approximations) can be carried out even when G is a grid graph (see Theorem 4).

Nevertheless, we bypass this hardness by designing $1 + \varepsilon$ approximation algorithms in time $f(\varepsilon, k, T) \cdot \text{poly}(n)$ (ie, EPASes in parameters k and T) for a broad class of structured input metrics, which includes (for example) the case when G is a planar graph. More precisely, we design approximation algorithms whenever input metric d_t has *bounded scatter dimension*. This concept was recently introduced by Abbasi *et al.* [ABB$^+$23], along with an algorithm demonstrating that metrics of bounded scatter dimension admit $f(\varepsilon, k) \cdot \text{poly}(n)$ time $1+\varepsilon$-approximation algorithms for many clustering problems. We generalize their ideas to obtain an EPAS for the (generalized) aggregate (k, z)-clustering problem. Bounded scatter dimension captures a rich class of metrics, including doubling metrics, shortest-path metrics on planar or minor-free graphs [BP25], and metrics of *bounded highway dimension*[4] — this last class, which was introduced to model transportation networks [AFGW10, ADF$^+$16], may be of particular interest in light of our road-network application from earlier. We emphasize that our lower bounds on the star and grid apply to metrics of bounded scatter dimension, so our FPT(k, T) approximation is essentially the best one could hope to achieve (up to the $1 + \varepsilon$ approximation) in this setting.

Additionally, we consider the case when G is a tree, or more generally a bounded-treewidth graph. As bounded-treewidth graphs have bounded scatter dimension, our $f(\varepsilon, k, T) \cdot \text{poly}(n)$-time algorithm from above applies. However, we can do even better — we obtain a $(1 + \varepsilon)$-approximation for the sum-aggregate (k, z)-clustering problem in time $f(T, \varepsilon) \cdot \text{poly}(n, k)$, by generalizing a folklore dynamic programming algorithm for k-median on bounded-treewidth graphs [ARR98, KR07, CFS21]. Parameterizing by T is necessary, because of our hardness result on the star. On the other hand, we can avoid parameterizing by k because our hardness for the $T = 3$ case only applies G is a grid graph, which has high treewidth — recall that graphs of bounded treewidth are precisely those graphs which exclude a grid as a minor [RS86, CT21]. In fact, we show in Theorem 4 that our $T = 3$ hardness rules out $f(T) \cdot \text{poly}(n)$-time algorithms for *any* graph G that has sufficiently large treewidth (specifically, $\Omega(\log^{295} n)$), whereas our upper bound in Theorem 36 achieves $f(T) \cdot \text{poly}(n)$ runtime so long as G has treewidth $o(\log n / \log \log n)$ (see Remark 37). Thus, our upper and lower bounds show that the treewidth of the base graph G almost entirely captures the tractability of FPT(T) algorithms for the sum-aggregate[5] clustering problem (up to some $1 + \varepsilon$ approximation factor, and up to some $\text{poly}(\log n)$ gap in the treewidth between our upper and lower bounds).

Open Directions We end this section mentioning some open directions for future study. If one observes the metrics we use to prove our hardness results, we notice that they "change a lot" as one moves from d_t to d_{t+1}. Can one obtain positive algorithmic results if this change is bounded? Concretely, assume d_t is the shortest path metric on $G_t = (V, E_t)$ and d_{t+1} is the same on $G_{t+1} = (V, E_{t+1})$. What if we forced $|E_t \Delta E_{t+1}| \leq O(1)$? Again taking the road-network example, these could lead to certain road-closures or road-building, but in a single period perhaps not too many edges are added or deleted. We do not know any lower bound for this version, nor do we know how to design algorithms. Structurally, this seems to require an understanding of how shortest paths change with insertion/deletion of edges which may be of independent interest.

Another interesting, but slightly less concrete, direction is to explore *online* algorithms where an algorithm

[4] We are not aware of a published proof that highway dimension has bounded scatter dimension, but the claim can be found at timestamp 0:52 in the recorded FOCS 2023 talk on [ABB$^+$23] (see link https://focs.computer.org/2023/schedule/) and also in [BGI25].

[5] We expect that a DP could be designed for other Ψ-aggregator norms, but in this paper we only consider $\Psi = \text{sum}$.

is allowed to change the centers but the distance metric is revealed only after the algorithms decision is made. Can one design "low-regret" algorithms as in [Zin03]? The sum-aggregate objective is in fact the benchmark with which regret is measured, and our results show that approximating this benchmark for general metrics is hard. With this in view, what would be a good definition of regret?

Coming back to aggregate clustering, a concrete open problem is the following. The aforementioned hardness when G is a star actually has $F \neq C$, but the case of $V = C = F$ is also interesting. Are there $f(T) \cdot \text{poly}(n, k)$ approximation or even exact algorithms for Ψ-aggregated k-supplier/median problem when G is a tree? We believe this is an interesting generalization of k-median problem on trees and worthy of exploration.

1.2 Related Works

As noted earlier, approximation algorithms for clustering problems have a rich history and we don't attempt to summarize this. Rather we mention the works most related to the "changing metric" viewpoint.

Deng, Li, and Rabani [DLR22] consider the k-clustering problems in a dynamic setting in which both clients and facilities may change over time (though the metric remains the same). The algorithm is allowed to change the open facilities over time, and the objective is to minimize the clustering costs and the distances that open facilities move over time. Their main results, at a high level, are in lockstep with our first set of results for aggregate clustering (on the tractability of $T = 2$ vs $T \geq 3$ scenarios); though our findings and theirs do not imply one another. Their setting has an ambient metric space (X, d) and, over T timesteps, there are client and facility sets $C_t, F_t \subseteq X$ for each timestep t. The goal is to pick k-sized solutions $S_t \subseteq F_t$ at each time step minimizing a certain objective function.

As mentioned earlier, our problem is related to stachastic/robust clustering problems which consider scenarios where client sets change but the algorithm needs to open the same set of k centers to serve these. This question was studied by Anthony, Goyal, Gupta, and Nagarajan [AGGN10] who obtain a $O(\log T + \log n)$-approximation for the robust k-median problem discussed earlier. They also show that stochastic k-center (i.e. stochastic k-supplier with $F = C$) is as hard to approximate as the densest-k-subgraph problem. Makarychev and Vakilian [MV21] obtain an $\left(e^{O(z)} \frac{\log T}{\log \log T}\right)$-approximation to robust (k, z)-clustering. Along with Chlamtáč, the same authors [CMV22] subsequently generalize this approximation algorithm to clustering with cascaded norm objectives, i.e. objectives of the form $\left(\sum_{t=1}^{T} \left(\sum_{j \in C_t} d(j, S)^z\right)^{q/z}\right)^{1/q}$. When $z \geq q$, they obtain a near-optimal $O\left(k^{O_{z,q}(1)}\right)$-approximation. When $z \leq q < \infty$, they obtain an $O_{z,q}(1)$-approximation.

Another related line of work is *dynamic* algorithms for clustering where clients come and go, and the algorithm needs to update the solution fast and maintain $O(1)$-approximations. There has been a lot of work [LV17, CGS18, CAHP+19, FLNFS21, BEF+23, BCLP23, ŁHG+24, FS25, BCLP25, BCF25] recently on these probelms. We point the readers to [BCLP25] (and references within) for dynamic k-center, and [BCF25] (and references within) for dynamic k-median. It is curious to see if techniques from these can address the questions mentioned in the open-directions paragraph earlier.

2 Hardness of Approximation

Theorem 3 (Hardness of approximation when $T \geq 3$)**.** *For any homogeneous aggregator Ψ and any $z \in \mathbb{N} \cup \{\infty\}$, there is no $f(T)\text{poly}(n, k)$-time finite-factor approximation for Ψ-aggregate (k, z)-clustering on finite metrics on n-vertices unless $P = NP$. The result holds even for $T = 3$.*

Proof. We show that it is NP-hard to decide whether the optimum value is finite or infinite for any Ψ-aggregate (k, z)-clustering problem. We reduce from the (perfect) 3D Matching problem where, recall, we are given a 3-partite hypergraph $H = (V_1 \uplus V_2 \uplus V_3, E)$ where $|V_1| = |V_2| = |V_3|$ any every hyperedge $e \in E$ intersects every V_t exactly once. The goal is to decide whether or not there exists a *matching* (pairwise disjoint subset) $M \subseteq E$ of hyperedges which spans all vertices. Equivalently, $|M \cap e| = 1$ for all $e \in E$.

Given such a 3D Matching instance $(V_1 \uplus V_2 \uplus V_3, E)$, we construct an aggregate clustering instance on $T = 3$ metrics, where the underlying graph is a complete graph G with vertex set E. The client set $C = F = E$ and $k \coloneqq |V_1| = |V_2| = |V_3|$. For each metric $t \in \{1, 2, 3\}$ and each two distinct clients e, e',

we set the distance $d_t(e, e') \coloneqq 0$ if $e \cap e' \cap V_t \neq \emptyset$, i.e. if they agree on their vertex in V_t. Otherwise set $d_t(e, e') \coloneqq \infty$. It is easy to see that each d_t is in fact a metric since if e_1, e_2 share the same V_t endpoint as do e_2, e_3, then they must be the same endpoint and thus e_1, e_3 share that as well. It remains to show that the clustering instance admits 0-cost solutions iff the original instance admits a perfect 3D matching.

- Suppose this clustering instance admits a 0-cost solution. Call this solution S. Fix a $t \in \{1, 2, 3\}$ and $v \in V_t$. Let $E(v)$ be the set of hyperedges that contain v. We know by construction that, $\forall e \in E(v), d_t(e, S) = 0$, i.e. $\exists e' \in S$ such that e, e' agree on their vertex in V_t. But then this vertex must be v itself. Thus $e' \in E(v) \cap S$, so $|E(v) \cap S| \geq 1$.

  Since $|V_t| = |S|$, this means that for each $v \in V_t$, $|E(v) \cap S| = 1$. So S is a perfect 3D matching.

- Suppose M is a perfect 3D matching in the original instance. Fix $t \in \{1, 2, 3\}$ and $e \in C$ such that $e \cap V_t = \{v\}$. We know that $M \cap E(v) \neq \emptyset$, so fix $e' \in M \cap E(v)$. Then $e \cap e' \cap V_t = \{v\}$, i.e. $d_t(e, e') = 0$, and so $d_t(e, M) = 0$. So for each $t \in \{1, 2, 3\}$ and each $e \in C$, we have $d_t(e, M) = 0$. Thus $\text{aggcost}(M) = 0$. □

Using techniques from graph drawing literature, one can show that our reduction Theorem 3 holds even when the base graph is a grid graph. Furthermore, we can rule out poly(n)-time algorithms for aggregate clustering whenever the base graph G contains even a relatively small $(\tilde{\Omega}(\log^3 n) \times \tilde{\Omega}(\log^3 n))$ grid minor, and consequently has relatively small treewidth; this follows from the fact that, under the randomized exponential time hypothesis, it is hard to obtain a $2^{o(n)}$-time algorithm for 3D matching [KBI19, Kus20]. We state this formally below, and provide the proof in Section 2.1.

Theorem 4. *Let G be any n-vertex graph with treewidth $\mathsf{tw} = \Omega(\log^{295} n)$. (For example, one may simply take G to be a $\sqrt{n} \times \sqrt{n}$ grid graph.) For any homogenous aggregator Ψ and any $z \in \mathbb{N} \cup \{\infty\}$, there is no $f(T)\text{poly}(n, k)$-time finite-factor approximation for Ψ-aggregate (k, z)-clustering problem when the base graph inducing the metrics is G, unless the randomized exponential time hypothesis is false. This results holds even for $T = 3$.*

Next, we give a different reduction to rule out that FPT algorithms parameterized by k, even when the base graph is a star.

Theorem 5 (Hardness of approximation on stars in FPT time.)**.** *For any homogenous aggregator Ψ and any $z \in \mathbb{N} \cup \{\infty\}$, there is no $f(k)\text{poly}(n, T)$-time finite-factor approximation algorithms for the Ψ-aggregate (k, z)-clustering problem even when the base graph G inducing the metrics is a star, unless W[2] = FPT.*

Proof. We reduce from k-hitting set, which is W[2] hard when parameterized by k [CFK$^+$15]. Suppose we are given an instance of k-hitting set defined by a universe $U = \{v_1, \ldots, v_n\}$ of n points, a set $\mathcal{X} = \{X_1, \ldots, X_m\}$ of m subsets of U, and an integer k; our task is to determine whether there are k points in U that hit all sets of $\mathcal{X}$. We construct an instance of aggregate clustering as follows. Let G be a star graph with a root vertex r connected to n leaves $v_1, \ldots, v_n$. Let the facility set be $F \coloneqq \{v_1, \ldots, v_n\}$, and let the client set (in every metric) be $C = \{r\}$. We define $T \coloneqq m$ distances functions, one for each set $X_t \in \mathcal{X}$, as follows: in the t-th metric, we set the weight of edge (r, v_i) to be 0 if $v_i \in X_t$ and ∞ otherwise.

It remains to show that the Ψ-aggregate (k, z)-clustering problem has finite cost iff there is a hitting set of size k.

- First we show that, if there is a hitting set S of size k, then the $\text{aggcost}(S) = 0$. Indeed, for every $t \in [T]$, there exists some $v_i \in S$ with $v_i \in X_t$ and so $d_t(r, S) \leq d_t(r, v_i) = 0$. In particular, the (k, z)-clustering cost paid by the client set $C = \{r\}$ in each metric is 0, and so the Ψ-aggregate cost is also 0 because Ψ is a norm.

- Similarly, if there is a set S of k facilities such that $\text{aggcost}(S) < \infty$, then we claim S is a hitting set. Indeed, consider some $X_t \in \mathcal{X}$. Since S has finite cost, we know that $d_t(r, S) < \infty$ and in particular there exists some $v_i \in S$ with $d_t(r, v_i) = 0$. By construction, v_i hits X_t. □

Remark 6. *The above reduction additionally shows that designing* bicriteria *approximation algorithms for aggregate clustering remains hard, due to the hardness of approximation of hitting set [Fei98] — there exists some $\beta = O(\log n)$ such that it is NP-hard to distinguish between the case that there is a hitting set of size k and the case that there is no hitting set of size βk.*

2.1 Proof of Theorem 4

We massage our reduction from Theorem 3. Observe that each of the $T = 3$ metrics used in the reduction Theorem 3 is a $0/\infty$ metric; that is, either points are at distance 0, or are at distance ∞. In particular, each metric can be realized as a disjoint union of paths (each with 0-weight edges), and so each metric is the shortest-path metric of a planar graph. However, it is not clear that there is an *underlying base graph* G that is planar, because vertices could belong to different paths in different metrics. We show that these sets of paths can be realized as reweightings of an underlying grid graph, or of any graph of sufficiently large treewidth; see Figure 1.

This reweighting scheme essentially follows from a much more general result by Schaefer [Sch21] about orthogonal drawings of planar graphs with fixed vertex locations. (Compare also to the classic theorem of Pach and Wenger [PW01] which proves something similar but without the guarantee of orthogonality.)

Lemma 7 (Theorem 9 of [Sch21])**.** *Let G be a planar graph with maximum degree at most 4, with n vertices $x_1, \ldots, x_n$. Let $p_1, \ldots, p_n$ be a set of n points in $\mathbb{R}^2$. Then there is an orthogonal drawing of G (ie, a drawing in which every edge is drawn as a combination of horizontal and vertical line segments) such that every vertex v_i is located at p_i, and every edge has at most $10n + 6$ bends. Moreover, this drawing can be found in $O(n^2)$ time.*

We adapt this lemma to our setting.

Lemma 8. *Let $n \in \mathbb{N}$, and let H be a sufficiently large $O(n^3) \times O(n^3)$ grid graph. There are n vertices $X = \{x_1, \ldots, x_n\}$ on H with the following property: for any $0/\infty$ metric d on the vertices X, there is a way to assign edge weights to H that induce a shortest-path metric d_H such that, for any $x_i, x_j \in X$, we have $d(x_i, x_j) = d_H(x_i, x_j)$. Moreover, this assignment of weights can be found in polynomial time.*

Proof. Given an arbitrary $0/\infty$ metric d on X, let $\{X_1, \ldots X_k\}$ denote a partition of X such that any two vertices x_i, x_j have $d(x_i, x_j) = 0$ iff they belong to the same part in the partition. Let G be a graph that connects vertices within each X_i by a path with 0-weight edges. Clearly G is planar and has maximum degree at most 2, and $d_G(x_i, x_j) = d(x_i, x_j)$ for all $x_i, x_j \in X$.

Let $\alpha = \Theta(n^2)$ with some sufficiently large hidden constant ($\alpha = 50n^2$ would suffice). For every $i \in [n]$, define $p_i \in \mathbb{R}^2$ by $p_i \coloneqq (i\alpha, i\alpha)$. By Lemma 7, there is an orthogonal drawing of G such every vertex x_i is drawn at location p_i. Now consider the set of "bends" in the drawing, ie the locations in $\mathbb{R}^2$ at which the drawing of some edge switches between being horizontal and being vertical. Observe that there are at most $(10n + 6)(3n - 6) = O(n^2)$ bends in total, as there are at most $3n - 6$ edges in a planar graph, and each edge has a drawing with at most $10n + 6$ bends. In particular, this means that for any two consecutive vertices x_i and x_{i+1}, there are at most most $O(n^2)$ bends with x-coordinate (resp. y-coordinate) between the x-coordinates (resp. y-coordinates) of x_i and x_{i+1}. As the x- (resp. y-) coordinates of x_i and x_{i+1} differ by $\alpha = \Theta(n^2)$, we may assume WLOG that the location of every bend has integer coordinates.

In other words, we may assume WLOG the drawing of G only uses edges of the $O(n\alpha) \times O(n\alpha)$ integer grid.

This drawing tells us how to reweight the grid graph H. To be precise, we assume that H is an $O(n\alpha) \times O(n\alpha)$ grid graph, and we let the vertex $(i\alpha, i\alpha) \in V(H)$ be the "designated location" for vertex $x_i \in X$. By our discussion above, any planar graph G can be drawn using edges of H. We define a weight function on $E(H)$, where edge $e \in E(H)$ has weight 0 if it is used in the drawing, and weight ∞ otherwise. Clearly $d_G(x_i, x_j) = d_H(x_i, x_j)$ for all $x_i, x_j \in X$ as desired. □

We observe that a similar reweighting can be carried out on any graph of high treewidth, because of the Excluded Grid theorem that connects treewidth to grid minors. The first version of the Excluded Grid theorem was proven by Robertson and Seymour [RS86] but with a much worse relationship between the treewidth and grid size; after a series of improvements, polynomial bounds were finally given by Chekuri and Chuzhoy [CC16]. (Note that there has been subsequent work that improves the polynomial [Chu15, Chu16, CT21], but their results are non-constructive.)

Lemma 9 (Excluded Grid theorem [CC16])**.** *If G is a graph with treewidth tw, then G contains a $(g \times g)$-grid as a minor for some $g = \Omega(\mathsf{tw}^{1/98}/\mathrm{poly}(\log \mathsf{tw}))$. Moreover, there is a randomized polynomial-time algorithm that finds the grid minor with high probability.*

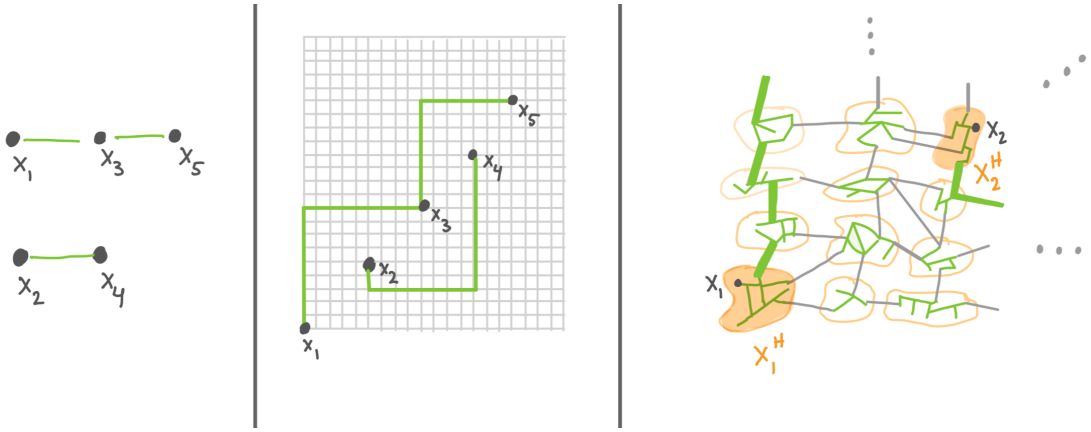

Figure 1: Left: A $0/\infty$-metric d represented as a planar graph. Center: A reweighting of a grid graph H that realizes d, as in Lemma 8. Right: A reweighting of a large-treewidth graph G that realizes d, as in Corollary 10; the supernodes of a grid minor are drawn in orange. In all three images, green lines represent 0-weight edges, and gray lines represent ∞-weight edges

Corollary 10. *Let $n \in \mathbb{N}$, and let G be a graph with treewidth $\mathsf{tw} = \Omega(n^{294} \cdot \mathrm{poly}(\log n))$ for some sufficiently large $\mathrm{poly}(\log n)$. There are n vertices $X = \{x_1, \ldots, x_n\}$ on G with the following property: for any $0/\infty$ metric $d(\cdot,\cdot)$ on the vertices X, there is a way to assign edge weights to G that induce a shortest-path metric d_G such that, for any $x_i, x_j \in X$, we have $d(x_i, x_j) = d_H(x_i, x_j)$. Moreover, there is a randomized $\mathrm{poly}(|V(G)|)$-time algorithm that finds these edge weights with high probability.*

Proof. By the Excluded Grid theorem, G contains a $(g \times g)$-grid minor H for $g = \Omega(n^3)$. By definition, there is a minor model of H in G; that is, there is a set $\mathcal{S}$ of disjoint *supernodes* (where each supernode is a connected subset of G) and a bijection $\psi : V(G) \to \mathcal{S}$ such that for any vertices in the grid $a, b \in V(H)$, the existence of an edge $(a, b) \in E(H)$ in the grid implies that there is some edge in G between a vertex in $\psi(a)$ and a vertex in $\psi(b)$. Moreover, there is a randomized algorithm that finds this minor model with high probability, in $\mathrm{poly}(|V(G)|)$ time.

By Lemma 8, we can select n designated vertices $X^H \subseteq V(H)$ such that any $0/\infty$ metric on X^H can be realized by reweighting H. Define $X \subseteq V(G)$ by selecting, for every $x_i^H \in X^H$, an arbitrary vertex x_i from the supernode $\psi(x_i^H)$.

Now suppose we are given some $0/\infty$ metric d on X. Equivalently, d could be viewed as a metric on X^H; under this view, let $w_H : E(H) \to \mathbb{R}$ denote the reweighting of H that realizes d, as guaranteed by Lemma 8. We define a reweighting $w : E(G) \to \mathbb{R}$ as follows. For any edge $e \in E(G)$, if e has both endpoints in the same supernode then $w(e) := 0$. If e has one endpoint in a supernode $\psi(a)$ and another in a supernode $\psi(b)$ for some $a, b \in V(H)$, and furthermore $(a, b) \in E(H)$, then set $w(e) := w_H((a, b))$. In all other cases, set $w(e) := \infty$. This reweighting satisfies $d_G(x_i, x_j) = d_H(x_i^H, x_j^H) = d(x_i, x_j)$ for all $x_i, x_j \in X$, as desired. □

For the proof of Theorem 4, we need one final ingredient, which is a refined perspective on the hardness of 3D matching. Recall that the *exponential time hypothesis* (ETH) is a standard assumption in fine-grained complexity that asserts there does not exist any algorithm that decides 3SAT instances with n literals in time $2^{o(n)}$. The *randomized exponential hypothesis* (rETH) asserts that this hardness holds even for randomized algorithms that decide 3SAT with high probability.

Lemma 11 ([KBI19, Kus20])**.** *If rETH holds, then there is no randomized algorithm that solves perfect 3D matching with high probability in time $2^{o(m)}$, where m is the number of hyperedges in the 3D matching instance.*

We are finally ready to prove Theorem 4.

Proof of Theorem 4. Let G be any graph with treewidth $\mathsf{tw} = \Omega(\log^{295} n)$. Let $(V_1 \uplus V_2 \uplus V_3, E)$ be an instance of perfect 3D matching with $m = \Theta(\log^{1.001} n)$ hyperedges. By Lemma 11, it is rETH-hard to

solve this instance in poly(n) time. As in Theorem 3, we define $T = 3$ metrics (each a $0/\infty$ metric) such that the aggregate clustering problem encode the 3D matching instance. By Corollary 10, each of these T metrics is the shortest path metric of some reweighting of G, and these reweightings can be found (with high probability) in poly(n) time. We conclude that aggregate clustering is hard even when the underlying graph is G. □

3 Constant-factor approximations for $T = 2$ scenarios

In this section we extend the vanilla $O(1)$-approximations for k-supplier and k-median for the case of $T = 2$ scenarios; this contrasts with the inapproximability for $T \geq 3$ (Theorem 3). For simplicity, we restrict our exposition when the aggregator function Ψ is just the sum, but we later point out in Remark 22 why it holds for any homogeneous aggregator such as a norm. In Remark 23 we explain how our algorithms generalize for (k, z)-clustering.

At a high level, both the extensions from vanilla work because (a very simple) matroid *intersection* is polynomial time tractable, while even 3D-matching is NP-hard; the latter was the root of the hardness for $T \geq 3$ aggregate clustering. We begin with the k-supplier problem which explains the previous line, and then show how ideas from the "matroid-median" problems solves the aggregate k-median problem for $T = 2$.

3.1 3-approximation for Aggregate k-Supplier

Let $\texttt{OPT}$ be the optimal set of k-suppliers which minimizes $\text{cost}_\infty(d_1; \texttt{OPT}) + \text{cost}_\infty(d_2; \texttt{OPT})$, where recall $\text{cost}_\infty(d_t; \texttt{OPT}) = \max_{j \in C} d_t(j, \texttt{OPT})$. Let us denote $\text{cost}_\infty(d_t; \texttt{OPT})$ as $\texttt{opt}_t$ for $t \in \{1, 2\}$. Note that this can be guessed in poly(n)-time (one could make this poly-logarithmic in n using "binary search" techniques) and so we assume we know $\texttt{opt}_1$ and $\texttt{opt}_2$. We now describe an algorithm, very similar to the one in [HS86], which returns a subset $\texttt{ALG}$ of facilities with $\text{cost}_\infty(d_t; \texttt{ALG}) \leq 3\texttt{opt}_t$ for $t \in \{1, 2\}$. This implies a 3-approximation for the aggregate k-supplier problem for the $T = 2$ case for any homogeneous aggregator Ψ.

For $t \in \{1, 2\}$, define $B_t(a, r) = \{b \in F \cup C : d_t(a, b) \leq r\}$. We run the Hochbaum-Shmoys [HS86] filtering algorithm on each metric d_t with parameter $\texttt{opt}_t$. To briefly describe this: (i) initially all clients are "uncovered", (ii) pick an arbitrary *uncovered client* $j \in C$ and add it to a set R_t of representatives, define $B_t(j, 2\texttt{opt}_t) \cap C$ to be j's "children", and mark j and all its children as "covered", and (iii) continue the above till all clients covered. The main observations, which are easy to check using triangle inequality and the fact that $\texttt{opt}_t$'s were correct guesses, are the following for $t \in \{1, 2\}$:

- The collection of balls $\{B_t(j, \texttt{opt}_t) \cap F\}_{j \in R_t}$ are pairwise disjoint.
- For any $j \in R_t$, $B_t(j, \texttt{opt}_t) \cap \texttt{OPT} \neq \emptyset$. The above two imply $|R_t| \leq k$.
- For every $v \in C$, $d_t(v, R_t) \leq 2\texttt{opt}_t$.

We now construct a solution $\texttt{ALG}$ of size $\leq k$ such that for any $t \in \{1, 2\}$ and any $j \in R_t$, we have $\texttt{ALG} \cap B_t(j, \texttt{opt}_t) \neq \emptyset$; that is, $\texttt{ALG}$ hits every ball in the above two collections. This can be solved using *matroid intersection* since there are only two collections of disjoint balls. Once we have such an $\texttt{ALG}$, it is easy to see using triangle inequality that, for any $t \in \{1, 2\}$, $d_t(v, \texttt{ALG}) \leq d_t(v, j) + d_t(j, \texttt{ALG}) \leq 3\texttt{opt}_t$ where j was v's representative in R_t.

To give more details on how to find $\texttt{ALG}$, for $t \in \{1, 2\}$ define the partition $\mathcal{P}_t$ of F formed by $\{B_t(j, \texttt{opt}_t) \cap F\}_{j \in R_t} \cup Z_t$, where Z_t are all the facilities of F not in any ball. We define the "budget" of each part to be $|\{B_t(j, \texttt{opt}_t) \cap F\}| - 1$ — this is the number of facilities $\texttt{ALG}$ can "leave out" — and set the budget of Z_t to be $|Z_t|$. We say $S \subseteq F$ is independent in partition matroid t if it picks at most budget from each part; we seek the set $S \subseteq F$ of the largest cardinality which is independent in both partition matroids. We return $\texttt{ALG} := F \setminus S$. By design, $\texttt{ALG}$ hits every $B_t(j, \texttt{opt}_t)$ for $t \in \{1, 2\}$ and $j \in R_t$. Since $F \setminus \texttt{OPT}$ is a candidate S which is in the intersection of the two partition matroids, we get $|\texttt{ALG}| = |F| - |S| \leq |F| - |F \setminus \texttt{OPT}| = |\texttt{OPT}| \leq k$.

In sum, we can find a solution $\texttt{ALG}$ such that $\text{cost}_\infty(d_t; \texttt{ALG}) \leq 3\texttt{opt}_t$ for $t \in \{1, 2\}$, and thus $\Psi(\text{cost}_\infty(d_1; \texttt{ALG}), \text{cost}_\infty(d_2; \texttt{ALG}) \leq 3\Psi(\texttt{opt}_1, \texttt{opt}_2)$. This completes the proof of the following theorem.

Theorem 12. *When $T = 2$, there is a* poly(n)*-time 3-approximation algorithm for Ψ-aggregate k-supplier on n vertices for any homogeneous aggregator Ψ .*

3.2 $O(1)$-approximation for Aggregate k-Median

To obtain an constant approximation for the k-median problem one has to work a bit harder, but the underlying idea behind tractability is still the tractability of matroid intersection. More precisely, it is the integrality of a polytope defined by two laminar set systems. At a high level, such an issue arises when one studies the *matroid median* problem where there is only one metric but one has the extra constraint that the set of facilities opened must be an independent set of a certain matroid. One solves this problem by rounding a solution to an LP-relaxation where matroid intersection (or rather the integrality fact mentioned above) forms a core component with the given matroid being one matroid, and the other formed via "filtering" technique a la [STA97, CGTS02]. In our case the situation is similar at such a high-level – the two "matroids" are formed by the filtering ideas for the $T = 2$ scenarios – but the details do need working out, and we show this below. We use the framework set by Swamy [Swa16] for the matroid-median problem, but other frameworks (such as the iterated rounding framework of [KLS18]) could possibly lead to better approximation factors; in this work, we didn't optimize the latter.

Linear Programming Relaxation We begin by writing a linear programming relaxation for our problem. Recall that, in the standard linear programs for k-median, matroid median, etc. [CGTS02, Li13, Swa16, DLR22], variables of the form $x(i, v)$ denote whether or not the client v uses the facility i, so that v's share of the cost is $\sum_{i\in F} d(v, i)x(i, v)$. In an integral solution $S \subseteq F$, each v has a unique $i_v \in S$ such that $x(i_v, v) = 1$, and $d(i_v, v) = d(v, S)$.

In our problem, we need *two* sets of such x-variables, because given an integral solution $S \subseteq F$, a client v can use different facilities under different metrics; that is, there can be distinct $i_1, i_2 \in S$ s.t. $d_1(v, S) = d_1(v, i_1)$ and $d_2(v, S) = d_2(v, i_2)$. So for each $t \in \{1, 2\}$, we define variables $\{x_t(i, v)\}_{v\in C, i\in F}$ denoting whether or not the client v uses the facility i under the metric d_t. We also have variables $\{y(i)\}_{i\in F}$ which, like in the standard k-median LP [CGTS02, CL12], denote whether or not the facility i is picked into our solution. This allows us to write linear constraints similar to the standard LP for k-median. We use lp to refer to the fractional optimum of the linear program relaxation.

$$\text{minimize: } \sum_{v\in C}\sum_{i\in F} d_1(i, v)x_1(i, v) + \sum_{v\in C}\sum_{i\in F} d_2(i, v)x_2(i, v) \tag{LP}$$

$$\sum_{i\in F} y(i) \le k \tag{LP1}$$

$$x_t(i, v) \le y(i) \qquad \forall t \in \{1, 2\}, v \in C, i \in F \tag{LP2}$$

$$\sum_{i\in F} x_t(i, v) = 1 \qquad \forall t \in \{1, 2\}, v \in C \tag{LP3}$$

$$0 \le x_t(i, v), y(i) \le 1 \qquad \forall t \in \{1, 2\}, v \in C, i \in F$$

Given a solution (x_1, x_2, y) of LP, we first make the following assumption which follows from *facility splitting* arguments due to Chudak and Shmoys [CS03] (see also [YC15])

Assumption 13. $\forall t \in \{1, 2\}, v \in C, i \in F$, $x_t(i, v) \in \{0, y(i)\}$.

We now perform a *filtering step* inherent to almost all LP-rounding algorithms for k-median. For our problem, as in the k-supplier problem from previous section, we get *two* sets of "representatives" instead of one that k-median rounding algorithms get.

Filtering. For $t \in \{1, 2\}$ and any client $v \in C$, define $C_t(v) := \sum_{i\in F} d_t(i, v)x_t(i, v)$, i.e. the cost paid by v under metric d_t. Our filtering algorithm begins with all clients in an *uncovered* set U. We pick $j \in U$ with the smallest $C_t(j)$, and call it a *representative under* d_t, adding it to the set Reps_t. Every *uncovered* client v that is within distance $4C_t(v)$ of j becomes a *child of* j *under* d_t, forming the set $\mathsf{child}_t(j)$. All of $\mathsf{child}_t(j)$ is then considered covered. We repeat this until all clients are covered. So we get $\mathsf{Reps}_t \subseteq C$ that is *well-separated*, i.e. any two distinct $j, j' \in \mathsf{Reps}_t$ have $d_t(j, j') > 4\max\{C_t(j), C_t(j')\}$. Reps_t also induces a partition $\{\mathsf{child}_t(j)\}_{j\in\mathsf{Reps}_t}$ of C. The following lemma is standard (see Section 3.3 for a proof), and thereafter, we focus on Reps_1 and Reps_2, and seek a constant-factor approximation on those clients only.

Lemma 14. *Consider $S \subseteq F$ and $\alpha \geq 1$, such that $\forall t \in \{1,2\}$,* $\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, d_t(j,S) \leq \alpha \cdot \sum_{v \in C} C_t(v)$. *Then* $\mathrm{aggcost}(S) \leq (4+\alpha) \cdot \mathtt{opt}$.

The rounding algorithms for many k-median algorithms first round to a half-integral solution and then to integral. We follow the same route, and in particular, follow Swamy's framework [Swa16]. We begin with the following definitions for each $t \in \{1,2\}, j \in \mathsf{Reps}_t$; as in the previous section, we use $B_t(a,r) = \{b \in F \cup C : d_t(a,b) \leq r\}$. An illustrative figure for these appears in Swamy's work [Swa16, Figure 1].

- $F_j^t := \left\{ i \in F : d_t(i,j) = \min_{j' \in \mathsf{Reps}_t} d_t(i,j') \right\}$ (breaking ties arbitrarily)
- $B_j^t := B_t(j, 2C_t(j)) \cap F$. Notice that, by construction of Reps_t, $B_j^t \subseteq F_j^t$ (also follows from Fact 15).
- $\gamma_j^{(t)} := \min_{i \notin F_j^t} d_t(i,j)$, and $G_j^t := \left\{ i \in F_j^t : d_t(i,j) \leq \gamma_j^{(t)} \right\}$.

We obtain the following a la [Swa16] (see Section 3.3 for a proof).

Fact 15. *For every $t \in \{1,2\}, j \in \mathsf{Reps}_t$, we have*

- $y(B_j^t) := \sum_{i \in B_j^t} y(i) \geq \frac{1}{2}$
- $B_j^t \subseteq G_j^t$
- *Suppose $\gamma_j^{(t)} = d_t(i_\gamma, j)$, s.t. $i_\gamma \in F_\ell^t$. Then $\forall i \in B_\ell^t, d_t(i,j) \leq 3\gamma_j^{(t)}$.*

Also note that by design the F_j^t's (and therefore the G_j^t's) are pairwise disjoint for a fixed t when we consider $j \in \mathsf{Reps}_t$. These play the role of the "partitions" as in the k-supplier problem, and our tractability follows because we have only two t's. Instead of finding the "largest cardinality independent set", as we did for the k-supplier problem, we instead find a point maximizing a suitable linear function. Towards this, we describe a "linear function" that acts as a proxy for $C_t(j)$'s.

For fractional facility masses $z \in [0,1]^F$, $\forall t \in \{1,2\}$, and $\forall j \in \mathsf{Reps}_t$, let $z(G_j^t) := \sum_{i \in G_j^t} z(i)$. Then we have

$$T_t(z,j) := \sum_{i \in G_j^t} d_t(i,j) z(i) + 3\gamma_j^{(t)} \max\left\{0, 1 - z(G_j^t)\right\}, \text{ and}$$
$$T_t(z) := \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, T_t(z,j)\,.$$

The above definition is quite similar to Swamy's definition which may be taken as the $T = 1$ case, with one difference that we have the "max with 0". When $T = 1$, one can assert $z(G_j) \leq 1$ but with different t's, it may be that $z(G_j^2) > 1$ because in metric 1 we need to open (fractionally) a lot of facilities. Nevertheless, even with the "max" function the following two lemmas similar to those in [Swa16] hold. The full proofs are in Section 3.3.

Lemma 16. *Consider $z \in [0,1]^F$ s.t. $\forall t \in \{1,2\}, j \in \mathsf{Reps}_t$, $z(B_j^t) \geq \frac{1}{2}$. Then j's assignment cost under d_t is at most $T_t(z,j)$. That is, for every $t \in \{1,2\}, j \in \mathsf{Reps}_t$, we can assign variables $\{0 \leq x'_t(i,j) \leq z(i)\}_{i \in F}$ so that $\sum_{i \in F} d_t(i,j) x'_t(i,j) \leq T_t(z,j)$.*

Lemma 17. *For each $t \in \{1,2\}$, and each $j \in \mathsf{Reps}_t$, $T_t(y,j) \leq 3 \cdot C_t(j)$.*

Rounding to Half-integral Solution. We are now ready to describe the algorithm which is encapsulated in the following lemma.

Lemma 18. *There is a polynomial time algorithm that yields a half-integral solution $(\hat{x}_1, \hat{x}_2, \hat{y})$ of LP s.t. $\forall t \in \{1,2\}$, $\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)| \sum_{i \in F} d_t(i,j) \hat{x}_t(i,j) \leq 6 \cdot \sum_{v \in C} C_t(v)$.*

Our goal now becomes to find a half-integral $\hat{y} \in \left\{0, \frac{1}{2}, 1\right\}^F$ so that, for each $t \in \{1,2\}$, $T_t(\hat{y}) \leq 6 \cdot \sum_{v \in C} C_t(v)$. If we are able to do so, then by Lemma 16 we would be done. To find such a half-intergral solution, we define the following polytope and linearization of T_t's. The polytope $\mathcal{P}$ has variables $(z, \lambda_1, \lambda_2) \in \mathbb{R}_{\geq 0}^{F \cup \mathsf{Reps}_1 \cup \mathsf{Reps}_2}$. The auxiliary λ_t variables serve to linearize the max terms in the $T_t(z)$'s, as the constraints in $\mathcal{P}$ ensure that $\lambda_t(j) \geq \max\left\{0, 1 - z(G_j^t)\right\}$.

$$\mathcal{P} := \Bigg\{ (z, \lambda_1, \lambda_2) \in \mathbb{R}_{\geq 0}^{F \cup \mathsf{Reps}_1 \cup \mathsf{Reps}_2} : z(F) \leq k;\ \forall t \in \{1,2\}, \forall j \in \mathsf{Reps}_t, \\ z(B_j^t) \geq \frac{1}{2},\ z(G_j^t) + \lambda_t(j) \geq 1 \Bigg\}$$

The following follows from the presence of only "two partitions" plus noting the λ's don't bother much since they are identity.

Claim 19. *$\mathcal{P}$ has half-integral extreme points.*

Proof. Consider the constraint matrix M of $\mathcal{P}$. Since the constant terms in the constraints are half-integral, it suffices [HK56] to show that M is totally unimodular (TU). Indeed, M looks $\left[\begin{array}{c|c} A & I \\ & 0 \end{array}\right]$ where we have the A-matrix corresponding to the z-variables, and the I corresponding to the λ_t-variables.

The dimension of I is $|\mathsf{Reps}_1| + |\mathsf{Reps}_2|$. The matrix A is the incidence matrix of two laminar systems — in fact, it is two partitions coarsened by the universe. Such a system is TU [Swa16]. A TU matrix padded with columns with at most one 1 in them remains TU. □

Over $\mathcal{P}$, we can then consider the *minimization* of the following linearization of the $T_t(z)$'s: $\forall t \in \{1,2\}, \forall j \in \mathsf{Reps}_t$: $W_t(z, \lambda_t, j) := \sum_{i \in G_j^t} d_t(i,j) z(i) + 3\gamma_j^{(t)} \lambda_t(j)$, and

$$W_t(z, \lambda_t) := \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, W_t(z, \lambda_t, j)\,.$$

The following claim encapsulates that we can find an extreme point of (any polytope) $\mathcal{P}$ which is a 2-approximation to both W_t's *simultaneously* in a sense made clear below. This allows us to get the desired $\hat{y}$ (as we explain better in the proof of Lemma 18).

Claim 20. *Consider a polytope $\mathcal{P} \subseteq \mathbb{R}_{\geq 0}^m$, linear functions $W_1, W_2 : \mathbb{R}_{\geq 0}^m \to \mathbb{R}_{\geq 0}$, and a point $p \in \mathcal{P}$. There is a polynomial time algorithm which returns an extreme point $\hat{p}$ of $\mathcal{P}$ such that $W_t(\hat{p}) \leq 2W_t(p)$ for $t \in \{1,2\}$.*

Proof. By Carathéodory's theorem [Car11, Ste13], we can write p as a linear combination of at most $(m+1)$ extreme points of $\mathcal{P}$. More precisely, we can find, in polynomial time, a subset $\mathcal{E}$ of extreme points such that $|\mathcal{E}| \leq m+1$ and $p = \sum_{q \in \mathcal{E}} \mu_q \cdot q$ for some μ_q's which form a probability distribution D_p. We claim that one of the $q \in \mathcal{E}$ is the desired point, and we can find which one by enumeration. To see why, observe that for $t \in \{1,2\}$, $\mathbb{E}_{\hat{p} \sim D_p}[W_t(\hat{p})] = W_t(p)$, and so by Markov, $\Pr_{\hat{p} \sim D_p}[W_t(\hat{p}) > 2W_t(p)] < \frac{1}{2}$. So, by union bound $\Pr_{\hat{p} \sim D_p}[\exists t \in \{1,2\} :\ W_t(\hat{p}) > 2W_t(p)] < 1$ implying $\Pr_{\hat{p} \sim D_p}[W_t(\hat{p}) \leq 2W_t(p),\ \ t \in \{1,2\}] > 0$ which proves the above claim. □

Proof of Lemma 18. Given our initial solution (x_1, x_2, y) of LP, set variables $\lambda_t(j) := \max\left\{0, 1 - y(G_j^t)\right\}$ for each $t \in \{1,2\}$, $j \in \mathsf{Reps}_t$. By Fact 15, this gives us $(y, \lambda_1, \lambda_2) \in \mathcal{P}$. So by Claim 19 and Claim 20, we can obtain a half-integral $(\hat{y}, \hat{\lambda}_1, \hat{\lambda}_2) \in \mathcal{P}$ such that $W_1(\hat{y}, \hat{\lambda}_1) \leq 2W_1(y, \lambda_1)$, and $W_2(\hat{y}, \hat{\lambda}_2) \leq 2W_2(y, \lambda_2)$. So by Lemma 16, for each $t \in \{1,2\}$ we can construct $\hat{x}_t$'s such that,

$$\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)| \sum_{i \in F} d_t(i,j) \hat{x}_t(i,j) \leq T_t(\hat{y}) \leq W_t(\hat{y}, \hat{\lambda}_t) \leq 2W_t(y, \lambda_t)$$

where the penultimate step follows from the constraints in $\mathcal{P}$. By our construction of the starting λ_t's, i.e. the constraints enforcing $\lambda_t(j) \geq \max\{0, 1 - z(G_j^t)\}$, $W_t(y, \lambda_t) = T_t(y)$. So we have, by Lemma 17, that the above can be upper bounded by $6 \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, C_t(j) \leq 6 \sum_{v \in C} C_t(v)$. □

Rounding to Integral solution. We round $(\hat{x}_1, \hat{x}_2, \hat{y})$ to an integral solution, losing a constant factor, and this is very similar to the ideas in [CGTS02, Swa16]; here also we follow the latter's framework.

Below are the main ideas, and the details are in Section 3.4.

The first step is another *filtering* step. For $t \in \{1,2\}$ and $j \in \mathsf{Reps}_t$, define $S_j^t := \{i \in F \ : \ \hat{x}_t(i,j) > 0\}$ and let $\hat{C}_t(j) := \sum_{i \in F} \hat{x}_t(i,j) d_t(i,j)$; note that $\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)| \hat{C}_t(j) \leq 6 \sum_{v \in C} C_t(v)$. Now we define a subset of "super-representatives": we pick $\ell \in \mathsf{Reps}_t$ with smallest $\hat{C}_t(\ell)$ and add it to Super_t removing every j with $S_j^t \cap S_\ell^t \neq \emptyset$ from consideration. In the end, we have S_j^t's pairwise disjoint for Super_t's and each $j \in \mathsf{Reps}_t$ shares a facility with a super-representative in Super_t.

Next, as in the rounding to the half-integral case, we define a particular linear function which works as a proxy for $\hat{C}_t(j)$. To define this proxy, for every $j \in \mathsf{Reps}_t$ call the two facilities in S_j^t primary and secondary in ascending order of distance (duplicating facilities if needed). For $t \in \{1,2\}$ and $j \in \mathsf{Reps}_t$, let $\ell \in \mathsf{Super}_t$ be the super-representative which j shares a facility with, and define

$$A_t(z,j) := \begin{cases} \sum_{i \in S_\ell^t} d_t(i,j) z(i) & \text{if } \mathsf{prim}_t(j) \in S_\ell^t \\ \sum_{i \in S_\ell^t} d_t(i,j) z(i) + (d_t(\mathsf{prim}_t(j), j) - d_t(\mathsf{sec}_t(j), j)) \cdot z(\mathsf{prim}_t(j)) & \text{otherwise} \end{cases}$$

It's instructive to think of all $z(i)$'s as $1/2$; in that case $A_t(z,j)$ is either the average of j's distance to ℓ's facilities when j's closest facility is shared with ℓ, or it is the average of j's distance to it closest facility and distance to the other facility that ℓ goes to whom j doesn't share. The importance of the above function is captured in a couple of observations: (A) for any t and any $j \in \mathsf{Reps}_t$, we can upper bound $A_t(\hat{y}, j) \leq 2\hat{C}_t(j)$, that is, $A_t(\hat{y}, j)$ isn't too bad a proxy, and perhaps more usefully (B) given any z such that $z(S_\ell^t) = 1$ for all *super representatives*, we can find an assignment of any $j \in \mathsf{Reps}_t$ to facilities with connection cost $\leq A_t(z,j)$. These two facts, and the fact that we can minimize linear functions over the intersection of two partition matroids (defined by the S_ℓ^t's for $t \in \{1,2\}$ and $\ell \in \mathsf{Super}_t$), gives us an $O(1)$-approximation. This is because the relevant polytope has integral extreme points (see Observation 28). Observation (A) holds because we can charge j's journey to ℓ's "other facility" to ℓ's connection cost which is smaller than j's cost by design; this uses half-intergrality. The details are in Lemma 27. Observation (B) is immediate if $\mathsf{prim}_t(j) \in S_\ell^t$ or if $z(\mathsf{prim}_t(j)) = 0$; otherwise, if $z(\mathsf{prim}_t(j)) = 1$ then j will travel to its closest facility while $A_t(z,j)$ will be average of two number which are larger than the minimum. The statement is in fact true for fractional z's as well and the proof is in Lemma 26.

All in all, we get the following theorem; the factor 28 could definitely be improved, but perhaps no better than 8 using these methods.

Theorem 21. *When $T = 2$, there is a polynomial time 28-approximation algorithm for sum-aggregate k-median.*

Remark 22 (Generalizing to arbitrary norm aggregators)**.** *In the above proof, we note that the solution S we return has the property that $\mathrm{cost}(d_t; S) \leq 28 \sum_{j \in C} C_t(j)$, for $t \in \{1,2\}$. This allows us to generalize the above theorem for any aggregator Ψ which is a norm. Instead of a linear program, we would have a* convex *program. More precisely, we have variables C_1, C_2 where $C_t := \sum_{j \in C} C_t(j)$ and the objective would minimize $\Psi(C_1, C_2)$. By the property of our rounding and the homogeneity of Ψ, we would get that $\Psi(\mathrm{cost}(d_1; S), \mathrm{cost}(d_2; S)) \leq 28\Psi(C_1, C_2) \leq 28\mathtt{opt}$.*

Remark 23 (Generalizing (k,z)-clustering)**.** *The above theorem focused on the k-median problem. However, the same methodology also gives an $O(1)$-approximation for the Ψ-aggregated (k,z)-clustering problem due to the fact that we only use triangle inequalities over "bounded number of hops". This is a folklore observation (see, for instance, Footnote 1 in [CS19]). To obtain this, first we replace $d_t(i,j)$ with $d_t(i,j)^z$ in the linear/convex program. Next, we use the "relaxed triangle inequality", which follows from Lemma A.1 in [MMR19], that says if we have $r+1$ points $a_1, \ldots, a_{r+1}$, then $d(a_1, a_{r+1})^z \leq r^{z-1} \sum_{i=1}^r d(a_i, a_{i+1})^z$. In our proof above, we never invoke the triangle inequality on more than 4 points, and thus, everything goes through with a "hit" of 3^{z-1}. Note, though, that the definition of the proxy function $T_t(z,j)$ would have 3 replaced by 3^z. In the end, we would get a $O(1)^z$ approximation to the sum of the zth powers, and since we take the zth root, this gives a $O(1)$-approximation.*

3.3 Deferred proofs towards half-integrality

Lemma 14. *Consider $S \subseteq F$ and $\alpha \geq 1$, such that $\forall t \in \{1,2\}$,* $\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, d_t(j,S) \leq \alpha \cdot \sum_{v \in C} C_t(v)$. *Then* $\mathrm{aggcost}(S) \leq (4+\alpha) \cdot \mathtt{opt}$.

Proof. For each $t \in \{1,2\}$, we have by construction that $d_t(v,j) \leq 4C_t(j) \leq 4C_t(v)$. Using this, we have

$$\begin{aligned}\sum_{v \in C} d_t(v,S) &= \sum_{j \in \mathsf{Reps}_t} \sum_{v \in \mathsf{child}_t(j)} d_t(v,S) \leq \sum_{j \in \mathsf{Reps}_t} \sum_{v \in \mathsf{child}_t(j)} (d_t(v,j) + d_t(j,S)) \\ &\leq \sum_{j \in \mathsf{Reps}_t} \sum_{v \in \mathsf{child}_t(j)} (4C_t(v)) + \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, d_t(j,S) \leq (4+\alpha) \cdot \sum_{v \in C} C_t(v)\,.\end{aligned}$$

So we have

$$\begin{aligned}\mathrm{aggcost}(S) &= \sum_{v \in C} d_1(v,S) + \sum_{v \in C} d_2(v,S) \\ &\leq (4+\alpha) \cdot \sum_{v \in C} C_1(v) + (4+\alpha) \cdot \sum_{v \in C} C_2(v) \\ &= (4+\alpha) \cdot \mathsf{lp} \leq (4+\alpha) \cdot \mathtt{opt}\,.\end{aligned}$$

□

Fact 15. *For every $t \in \{1,2\}, j \in \mathsf{Reps}_t$, we have*

- $y(B_j^t) := \sum_{i \in B_j^t} y(i) \geq \frac{1}{2}$
- $B_j^t \subseteq G_j^t$
- *Suppose $\gamma_j^{(t)} = d_t(i_\gamma, j)$, s.t. $i_\gamma \in F_\ell^t$. Then $\forall i \in B_\ell^t, d_t(i,j) \leq 3\gamma_j^{(t)}$.*

Proof. Fix $t \in \{1,2\}$ and $j \in \mathsf{Reps}_t$.

- If $C_t(j) = 0$, then $y(B_j^t) = 1$. Otherwise,

$$\begin{aligned}C_t(j) &\geq \sum_{i \notin B_j^t} x_t(i,j) d_t(i,j) > 2C_t(j) \cdot \left(\sum_{i \notin B_j^t} x_t(i,j) \right) \\ &= 2C_t(j) \cdot \left(1 - \sum_{i \in B_j^t} x_t(i,j) \right) \geq 2C_t(j) \cdot \left(1 - y(B_j^t)\right)\end{aligned}$$

  which, when $C_t(j) > 0$, gives $y(B_j^t) \geq \frac{1}{2}$.

- By construction of the child_t sets, $B_j^t \in F_j^t$, so it suffices to show that $\gamma_j^{(t)} \geq 2C_t(j)$. For this, suppose $\gamma_j^{(t)} = d_t(i_\gamma, j)$, $i_\gamma \in F_\ell^t$. Then $\gamma_j^{(t)} \geq d_t(j,\ell) - d_t(i_\gamma,\ell) \geq d_t(j,\ell) - \gamma_j^{(t)}$. So $2\gamma_j^{(t)} \geq d_t(j,\ell) > 4C_t(j)$ by the construction of child_t sets; i.e. $\gamma_j^{(t)} \geq 2C_t(j)$.
- We have $d_t(i,j) \leq d_t(j,\ell) + d_t(i,\ell)$. We also have, by the construction of the child_t sets, that that $d_t(i,\ell) \leq 2C_t(\ell) \leq 2\max\{C_t(\ell), C_t(j)\} \leq \frac{1}{2} \cdot d_t(\ell,j)$. So $d_t(i,j) \leq \frac{3}{2} \cdot d_t(j,\ell)$. Finally, since $i_\gamma \in F_\ell^t$, we have $d_t(i_\gamma,\ell) \leq d_t(i_\gamma,j)$, and hence $d_t(j,\ell) \leq d_t(i_\gamma,j) + d_t(i_\gamma,\ell) \leq 2d_t(i_\gamma,j) = 2\gamma_j^{(t)}$. So $d_t(i,j) \leq 3\gamma_j^{(t)}$.

□

Lemma 16. *Consider $z \in [0,1]^F$ s.t. $\forall t \in \{1,2\}, j \in \mathsf{Reps}_t$, $z(B_j^t) \geq \frac{1}{2}$. Then j's assignment cost under d_t is at most $T_t(z,j)$. That is, for every $t \in \{1,2\}, j \in \mathsf{Reps}_t$, we can assign variables $\{0 \leq x'_t(i,j) \leq z(i)\}_{i \in F}$ so that $\sum_{i \in F} d_t(i,j) x'_t(i,j) \leq T_t(z,j)$.*

Proof. If $z(G_j^t) \geq 1$, then we can set $x'_t(i,j)$'s such that $\sum_{i \in G_j^t} x'_t(i,j) = 1$ and $\sum_{i \notin G_j^t} x'_t(i,j) = 0$. This gives the assignment cost

$$\sum_{i \in F} d_t(i,j) x'_t(i,j) = \sum_{i \in G_j^t} d_t(i,j) x'_t(i,j) \leq \sum_{i \in G_j^t} d_t(i,j) z(i) \leq T_t(z,j)\,.$$

Otherwise, we set $x'_t(i,j) = z(i)$ for every $i \in G_j^t$. Then, we identify $\ell \in \mathsf{Reps}_t$ s.t. $\gamma_j^{(t)} = d_t(i',j)$ for some $i' \in F_\ell$. Fact 15 gives us that $z(B_\ell^t) \geq \frac{1}{2} \geq 1 - z(B_j^t) \geq 1 - z(G_j^t) = 1 - \sum_{i \in G_j^t} x'_t(i,j)$. So we can set $x'_t(i,j)$'s s.t. they are zero outside $G_j^t \cup B_\ell^t$, and $\sum_{i \in B_\ell^t} x'_t(i,j) = 1 - \sum_{i \in G_j^t} z(i)$. This gives us the assignment cost

$$\begin{aligned}
\sum_{i \in F} d_t(i,j) x'_t(i,j) &= \sum_{i \in G_j^t} d_t(i,j) x'_t(i,j) + \sum_{i \in B_{j'}^t} d_t(i,j) x'_t(i,j) \\
&\leq \sum_{i \in G_j^t} d_t(i,j) z(i) + 3\gamma_j^{(t)} \left(\sum_{i \in B_{j'}^t} x'_t(i,j) \right) \\
&= \sum_{i \in G_j^t} d_t(i,j) z(i) + 3\gamma_j^{(t)} \left(1 - z(G_j^t)\right) = T_t(z,j)\,.
\end{aligned}$$

□

Lemma 17. *For each $t \in \{1,2\}$, and each $j \in \mathsf{Reps}_t$, $T_t(y,j) \leq 3 \cdot C_t(j)$.*

Proof. If $y(G_j^t) \geq 1$, then we have that

$$C_t(j) = \sum_{j \in \mathsf{Reps}_t} x_t(i,j) d_t(i,j) = \sum_{j \in \mathsf{Reps}_t} d_t(i,j) y(i) = T_t(y,j)\,.$$

Otherwise, we can say by Assumption 13 that

$$\begin{aligned}
C_t(j) &= \sum_{i \in G_j^t} d_t(i,j) x_t(i,j) + \sum_{i \notin G_j^t} d_t(i,j) x_t(i,j) \\
&\geq \sum_{i \in G_j^t} d_t(i,j) y(i) + \gamma_j^{(t)} \cdot \left(1 - \sum_{i \in G_j^t} x_t(i,j)\right) && \ldots \text{by (LP2)} \\
&= \sum_{i \in G_j^t} d_t(i,j) y(i) + \gamma_j^{(t)} \left(1 - y(G_j^t)\right) \geq \frac{1}{3} \cdot T_t(y,j)\,.
\end{aligned}$$

□

3.4 Rounding to an integral solution

Given the half-integral solution $(\hat{x}_1, \hat{x}_2, \hat{y})$, we first make the assumption below.

Assumption 24. $\forall i \in F$, $\hat{y}(i) \in \left\{0, \frac{1}{2}\right\}$.

This assumption can be made true by splitting any $i \in F$ s.t $\hat{y}(i) = 1$ into two copies i', i'' with $\hat{y}(i') = \hat{y}(i'') = \frac{1}{2}$; and then reassigning $\hat{x}_t$'s accordingly. This then implies Assumption 13 as well.

Filtering. We now perform another simpler filtering step in each metric. For $t \in \{1,2\}$ and $j \in \mathsf{Reps}_t$, we define $S_j^t := \{i \in F : \hat{x}_t(i,j) > 0\}$, and $\hat{C}_t(j) := \sum_{i \in F} \hat{x}_t(i,j) d_t(i,j)$. We begin with all of Reps_t being in a set U of *uncovered* representatives. We pick $\ell \in U$ with the smallest $\hat{C}_t(\ell)$, and call it a *super-representative under d_t*, adding it to the set Super_t. We define its *neighbor set* $\mathsf{nbr}_t(\ell) := \left\{j \in \mathsf{Reps}_t : S_j^t \cap S_\ell^t \neq \emptyset\right\}$. We consider $\mathsf{nbr}_t(\ell)$ covered, and then repeat this process until all of Reps_t is covered.

Unlike in the previous phase, we do not discard Reps_t after forming Super_t. We open facilities near the super-representatives, but still consider the assignment cost of all representatives.

Rounding. For a $t \in \{1,2\}$, and $j \in \mathsf{Reps}_t$, we define $\mathsf{prim}_t(j)$ to be the facility in S_j^t that is closer to j, and $\mathsf{sec}_t(j)$ to be the other one. Now suppose $j \in \mathsf{nbr}_t(\ell)$ for an $\ell \in \mathsf{Super}_t$. We define, for a $z \in [0,1]^F$,

$$A_t(z,j) := \begin{cases} \sum_{i \in S_\ell^t} d_t(i,j) z(i) & \text{if } \mathsf{prim}_t(j) \in S_\ell^t \\ \sum_{i \in S_\ell^t} d_t(i,j) z(i) + (d_t(\mathsf{prim}_t(j), j) - d_t(\mathsf{sec}_t(j), j)) \cdot z(\mathsf{prim}_t(j)) & \text{otherwise} \end{cases}$$

and $A_t(z) = \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, A_t(z,j)$. $A_t(z)$ serves as a linear proxy for the cost under metric d_t. We observe, directly from LP3, that

Observation 25. $\forall t \in \{1,2\}, j \in \mathsf{Reps}_t$, $\hat{y}(S_j^t) = 1$.

We then obtain bounds analogous to Lemmas 16 and 17.

Lemma 26. *Consider a $z \in [0,1]^F$ s.t. for each $t \in \{1,2\}$, and $\ell \in \mathsf{Super}_t$, we have $z(S_\ell^t) = 1$. Then the assignment cost of j under d_t is at most $A_t(z,j)$. That is, we can assign variables $\{0 \le x'_t(i,j) \le z(i)\}_{i \in F}$ such that, for each $t \in \{1,2\}$ and $j \in \mathsf{Reps}_t$, $\sum_{i \in F} x'_t(i,j) d_t(i,j) \le A_t(z,j)$.*

Proof. If $z(\mathsf{prim}_t(j)) = 0$, or if $\mathsf{prim}_t(j) \in S_\ell^t$, then $A_t(z,j) = \sum_{i \in S_\ell^t} d_t(i,j) z(i)$. In these cases, set $x'_t(i,j) = z(i)$ for each $i \in S_\ell^t$, and other $x'_t(i,j)$'s to zero.

Now consider the case where $\mathsf{prim}_t(j) \notin S_\ell^t$, and $z(\mathsf{prim}_t(j)) > 0$. Set $x'_t(\mathsf{prim}_t(j), j) = z(\mathsf{prim}_t(j))$. By construction of Super_t, we must have $\mathsf{sec}_t(j) \in S_\ell^t$. Let the other facility in S_ℓ^t be i_0. Note that since $\hat{x}_t(i_0, j) < \hat{x}_t(\mathsf{sec}_t(j))$, we can assume that $d_t(\mathsf{sec}_t(j), j) \le d_t(i_0, j)$; because if not, then swapping the values of $\hat{x}_t(i_0, j)$ and $\hat{x}_t(\mathsf{sec}_t(j))$ can only decrease the assignment cost. We use this in both subcases below.

- If $z(S_j^t) \ge 1$, set $x'_t(\mathsf{sec}_t(j), j) = 1 - z(\mathsf{prim}_t(j))$, and remaining $x'_t(i,j)$'s to zero. The assignment cost becomes $d_t(\mathsf{prim}_t(j), j) z(\mathsf{prim}_t(j)) + d_t(\mathsf{sec}_t(j), j)(1 - z(\mathsf{prim}_t(j)))$. Since $z(S_\ell^t) = 1$, we can write the second term as

$$\begin{aligned} & d_t(\mathsf{sec}_t(j), j) \cdot (z(i_0) + z(\mathsf{sec}_t(j)) - z(\mathsf{prim}_t(j))) \\ \le & d_t(i_0, j) z(i_0) + d_t(\mathsf{sec}_t(j), j) z(\mathsf{sec}_t(j)) - d_t(\mathsf{sec}_t(j)) z(\mathsf{prim}_t(j)) \\ = & \sum_{i \in S_\ell^t} d_t(i,j) z(i) - d_t(\mathsf{sec}_t(j)) z(\mathsf{prim}_t(j)) \,. \end{aligned}$$

  Adding back the first term, the overall assignment cost is then at most $A_t(z,j)$.

- If $z(S_j^t) < 1$, set $x'_t(\mathsf{sec}_t(j), j) = z(\mathsf{sec}_t(j))$, $x'_t(i_0, j) = 1 - z(\mathsf{prim}_t(j)) - z(\mathsf{sec}_t(j))$, and remaining $x'_t(i,j)$'s to zero. Note that, since $z(S_\ell^t) = 1$, $x'_t(i_0, j) = z(i_0) - z(\mathsf{prim}_t(j)) \le z(i_0)$, as required. We first note that

$$\begin{aligned} & d_t(i_0, j) \cdot (z(i_0) - z(\mathsf{prim}_t(j)) - z(\mathsf{sec}_t(j))) \\ \le & d_t(i_0, j) z(i_0) - d_t(\mathsf{sec}_t(j), j) z(\mathsf{prim}_t(j)) - d_t(\mathsf{sec}_t(j), j) z(\mathsf{sec}_t(j)) \,. \end{aligned}$$

  Adding this to $d_t(\mathsf{prim}_t(j)) z(\mathsf{prim}_t(j)) + d_t(\mathsf{sec}_t(j)) z(\mathsf{sec}_t(j))$, the assignment cost becomes

$$d_t(i_0, j) z(i_0) + (d_t(\mathsf{prim}_t(j), j) - d_t(\mathsf{sec}_t(j), j))\, z(\mathsf{prim}_t(j)) \le A_t(z,j) \,.$$

□

Lemma 27. *Given half-integral facility masses $\hat{y} \in \left\{0, \frac{1}{2}, 1\right\}^F$, for each $t \in \{1,2\}$ and each $j \in \mathsf{Reps}_t$, $A_t(\hat{y}, j) \le 2\hat{C}_t(j)$.*

Proof. Let j be in $\mathsf{nbr}_t(\ell)$. Via Assumption 24, we have $\hat{y} \in \left\{0, \frac{1}{2}\right\}^F$.

In the case $\mathsf{prim}_t(j) \in S^t_\ell$, let i_0 be the other facility in S^t_ℓ. Then,

$$\begin{aligned}
A_t(\hat{y}, j) &= \frac{1}{2}\left(d_t(\mathsf{prim}_t(j), j) + d_t(i_0, j)\right) \le \frac{1}{2}\left(2d_t(\mathsf{prim}_t(j), j) + d_t(\mathsf{prim}_t(j), \ell) + d_t(i_0, \ell)\right) \\
&\le d_t(\mathsf{prim}_t(j), j) + \hat{C}_t(\ell) \le \frac{1}{2}\left(d_t(\mathsf{prim}_t(j), j) + d_t(\mathsf{sec}_t(j), j)\right) + \hat{C}_t(\ell) \\
&= \hat{C}_t(j) + \hat{C}_t(\ell) \le 2\hat{C}_t(j)\,.
\end{aligned}$$

In the case $\mathsf{prim}_t(j) \notin S^t_\ell$, we know $\mathsf{sec}_t(j) \in S^t_\ell$. Once again, let i_0 be the other facility in S^t_ℓ. Then,

$$\begin{aligned}
A_t(\hat{y}, j) &= \frac{1}{2}\left(d_t(\mathsf{sec}_t(j), j) + d_t(i_0, j)\right) + \frac{1}{2}\left(d_t(\mathsf{prim}_t(j), j) - d_t(\mathsf{sec}_t(j), j)\right) \\
&= \frac{1}{2}\left(d_t(i_0, j) + d_t(\mathsf{prim}_t(j), j)\right) \\
&\le \frac{1}{2}\left(d_t(i_0, \ell) + d_t(\ell, \mathsf{sec}_t(j)) + d_t(\mathsf{sec}_t(j), j) + d_t(\mathsf{prim}_t(j), j)\right) \\
&= \hat{C}_t(\ell) + \hat{C}_t(j) \le 2\hat{C}_t(j)\,.
\end{aligned}$$

□

Now, we want an integral $\tilde{y} \in \{0,1\}^F$ s.t. for each $t \in \{1,2\}$,
$A_t(\tilde{y}) \le O(1) \cdot \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)|\, \hat{C}_t(j)$. We define the following polytope:

$$\mathcal{R} := \left\{ z \in \mathbb{R}^F_{\ge 0} : z(F) \le k;\, \forall t \in \{1,2\}, \forall \ell \in \mathsf{Super}_t, z(S^t_\ell) = 1 \right\}.$$

Since the constraints in $\mathcal{R}$ are over two laminar families, its constraint matrix is totally unimodular. The constant terms in the constraints are all integral, so we have via properties of total unimodularity [HK56] that

Observation 28. *$\mathcal{R}$ has integral extreme points.*

The remaining algorithm is analogous to the previous phase.

Lemma 29. *There is a polynomial time algorithm which returns an integral solution $(\tilde{x}_1, \tilde{x}_2, \tilde{y})$ of LP s.t. $\forall t \in \{1,2\}$, $\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)| \sum_{i \in F} d_t(i,j)\tilde{x}_t(i,j) \le 24 \cdot \sum_{v \in C} C_t(v)$.*

Proof. Let $\hat{y}$ be the half-integral solution obtained via Lemma 18.

Now for a $j \in \mathsf{Reps}_t$, by Claim 20 and Observation 28, we can obtain an integral $\tilde{y} \in \mathcal{R}$ such that $A_1(\tilde{y}) \le 2A_1(\hat{y})$ and $A_2(\tilde{y}) \le 2A_2(\hat{y})$. This is because, in Claim 20, we can plug in A_1, A_2 to be our linear functions, $\mathcal{R}$ to be our polytope, $\hat{y}$ to be our starting point p, and $\tilde{y}$ to be the sampled extreme point. So by Lemmas 26 and 27, we can construct $\tilde{x}_t$'s s.t., for any $\varepsilon' > 0$,

$$\begin{aligned}
&\sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)| \sum_{i \in F} d_t(i,j)\tilde{x}_t(i,j) \le A_t(\tilde{y}) \le 2A_t(\hat{y}) \\
&\le 2 \cdot 2 \sum_{j \in C} |\mathsf{child}_t(j)|\, \hat{C}_t(j) = 4 \sum_{j \in \mathsf{Reps}_t} |\mathsf{child}_t(j)| \sum_{i \in F} d_t(i,j)\hat{x}_t(i,j) \\
&\le 4 \cdot 6 \cdot \sum_{v \in C} C_t(v) \le 24 \cdot \sum_{v \in C} C_t(v)
\end{aligned}$$

where the penultimate step is from Lemma 18. □

This, combined with Lemma 14, means that we obtain an integral solution
$S := \{i \in F : \tilde{y}(i) = 1\}$ such that $\mathrm{cost}_1(d_1; S) + \mathrm{cost}(d_2; S) \le 28 \cdot \mathsf{lp}$. This proves Theorem 21.

4 FPT(k, T) constant factor approximation

Theorem 30. *For any homogenous aggregator Ψ, any $z \in \mathbb{N} \cup \{\infty\}$, and any constant error parameter $\varepsilon > 0$, there is an algorithm that computes a $(3+\varepsilon)$-approximate solution to Ψ-aggregate (k, z)-clustering in time $O(\varepsilon^{-1}k)^{O(kT)} \cdot 2^{O(k^2T^2)} \cdot \mathrm{poly}(n)$.*

Proof. We adapt an existing FPT 3-approximation [CAGK+19, Section 2.1] for standard (k, z)-clustering (i.e. $T = 1$). Fix an error parameter $\varepsilon > 0$.

In our aggregate cluster instance, let C denote the client set and let F denote the facilities. For each scenario $t \in [T]$ (defined by a metric d_t and a client weights w_t), we first compute a *coreset* $C_t \subseteq C$ of size $O(\varepsilon^{-2}k \log n)$ [FL11]; that is, obtain a new client set C_t and new weights w'_t on C_t with the property that for any subset of facilities $S \subseteq F$,

$$\sum_{u \in C_t} w'_t(u) \cdot d_t(u, S) \in (1 \pm \varepsilon) \cdot \sum_{v \in C} w_t(v) \cdot d_t(v, S)\,.$$

Henceforth we work with the coresets C_t rather than the entire client set C in each metric.

Let $\texttt{OPT} \subseteq F$ be the optimal solution to the aggregate clustering instance, with (unknown) centers $o_1, \ldots, o_k$. For an $i \in [k]$ and $t \in [T]$, let $C_{i,t}$ be the cluster served by o_i under d_t. Define the client $\ell_{i,t}$ to be the closest client (in the coreset C_t) to o_i, that is $d_t(o_i, C_{i,t}) = d_t(o_i, \ell_{i,t})$, and define $r_{i,t} \coloneqq d_t(o_i, \ell_{i,t})$.

For each scenario $t \in [T]$ and index $i \in [k]$, we can guess (by enumeration) the client $\ell_{i,t}$ in time $\binom{O(\varepsilon^{-2}k \log n)}{k}$. We can't guess $r_{i,t}$ exactly, but we can guess it up to the closest power of $(1+\varepsilon)$: we assume WLOG (up to a $1+\varepsilon$ loss in the approximation factor) that the aspect ratio of d_t is $\mathrm{poly}(n)$, and so bucketing pairwise distances of d_t by powers of $(1+\varepsilon)$ reduces the number of possible distances to $O(\log_{1+\varepsilon} n) = O(\varepsilon^{-1} \log n)$. Overall, in time $(\binom{O(\varepsilon^{-2}k \log n)}{k} \cdot O(\varepsilon^{-1} \log n)^k)^T$, we can guess each $\ell_{i,t}$ and $r_{i,t}$. By Fact 39, this overhead is $O(\varepsilon^{-1}k)^{O(kT)} \cdot 2^{O(k^2T^2)} \cdot \mathrm{poly}(n) = O_{\varepsilon,k,T}(1) \cdot \mathrm{poly}(n)$.

Suppose our guesses are correct. For each $i \in [k]$, consider the set F_i containing the facilities that are within distance $r_{t,t}$ of $\ell_{i,t}$ in every metric d_t, i.e. $F_i \coloneqq \cap_{t=1}^{T} B_t(\ell_{i,t}, r_{i,t}) \cap F$. If our guesses are correct, then F_i is non-empty since o_i lies in this intersection. Let our solution $\texttt{ALG}$ consist of an arbitrary $f_i \in F_i$ for each $i \in [k]$. For a client v in the cluster $C_{i,t}$, we have $d_t(o_i, \ell_{i,t}) \le d_t(v, o_i)$ by definition of $\ell_{i,t}$; and also $d_t(\ell_{i,t}, f_i) \le (1 + O(\varepsilon)) \cdot r_{i,t} = (1 + O(\varepsilon)) \cdot d_t(\ell_{i,t}, o_i)$. Using these, we get

$$\begin{aligned} d_t(v, \texttt{ALG}) &\le d_t(v, f_i) \le d_t(v, o_i) + d_t(o_i, \ell_{i,t}) + d_t(\ell_{i,t}, f_i) \\ &\le d_t(v, o_i) + (2 + O(\varepsilon)) \cdot d_t(o_i, \ell_{i,t}) \le (3 + O(\varepsilon)) \cdot d_t(v, o_i) \end{aligned}$$

for every $v \in C$ and $t \in [T]$. Since the (k, z)-clustering cost and the aggregator Ψ are both homogeneous functions, this yields a $(3 + O(\varepsilon))$-approximation.

Rescaling ε by a constant yields the desired result. □

Simpler algorithm in a special case

In the special case of k-supplier, we have a simpler $(3+\varepsilon)$-algorithm that doesn't utilize coresets, and runs in $O(\varepsilon^{-O(T^2)}) \cdot O(k)^{O(kT)} \cdot \mathrm{poly}(n)$ time. This algorithm does not require coresets for its enumeration component.

Guessing radii. We use the same bucketing as above, which contributes a $(1+\varepsilon)$ factor and allows us to assume that there are $O\left(\frac{\log n}{\varepsilon}\right)$ distances per metric.

Now let $\texttt{opt}_t \coloneqq \mathrm{cost}(d_t; S)$ for each $t \in [T]$, and note that each $\texttt{opt}_t$ is some $d_t(u, v)$ times either $w_t(v)$ or $w_t(u)$. So with $O\left(\frac{\log n}{\varepsilon}\right)^T$ overhead, which due to Fact 39 is an $\varepsilon^{-O(T^2)} \cdot \mathrm{poly}(n)$ overhead, we can guess $\texttt{opt}_t$ in each metric. Then for each $v \in V$ and $t \in [T]$, let $r_t(v) \coloneqq \texttt{opt}_t / w_t(v)$ be the radius within which, under d_t, the client v must be served.

Guessing clusters. For each $t \in [T]$, we run Plesník's [Ple87] algorithm in each metric d_t with radii $\{r_t(v)\}_{v \in V}$. These Plesník subroutines yield, for each $t \in [T]$:

- Clusters $C_{1,t}, \ldots, C_{k'_t,t} \subseteq C$ for some $k'_t \le k$;

- A center $h_{i,t} \in C_{i,t}$ for each cluster, such that $C_{i,t} \subseteq B_t(h_{i,t}, 2r_t(h_{i,t}))$ and $\forall v \in C_{i,t}$, $r_t(v) \geq r_t(h_{i,t})$;
- For any two distinct clusters $C_{i,t}$ and $C_{j,t}$, $d_t(h_{i,t}, h_{j,t}) > r_t(h_{i,t}) + r_t(h_{j,t})$.

Thereafter, for each $t \in [T]$, we augment the clusters above with empty clusters $C_{k'_t+1,t}, \ldots, C_{k+1}$; and call the resulting partition $\mathcal{C}_t$. Note that each $\mathcal{C}_t$ has at most k non-empty clusters.

Suppose the (unknown) centers in $\texttt{OPT}$ are $o_1, \ldots, o_k$. With $(k+1)!^T$ overhead, we can guess one cluster per metric whose center is served by o_i; and rename WLOG so that this cluster is called $C_{i,t}$. That is, for each $i \in [k]$ and $t \in [T]$, we guess $C_{i,t} \in \mathcal{C}_t$ such that $d_t(h_{i,t}, \texttt{OPT}) = d_t(h_{i,t}, o_i)$. This is possible to guess in the desired runtime because any nonempty $C_{i,t}$ is unique in $\mathcal{C}_t$; to see this, suppose not, and consider the distinct centers $h_{i,t}, h_{j,t}$ served by o_i under d_t. Then $d_t(h_{i,t}, h_{j,t}) \leq d_t(h_{i,t}, o_i) + d_t(h_{j,t}, o_i) \leq r_t(h_{i,t}) + r_t(h_{j,t})$, violating one of the Plesník properties listed above.

Picking centers. For each $i \in [k]$, let $F_i = F \cap \bigcap_{t \in [T]: C_{i,t} \neq \emptyset} B_t(h_{i,t}, r_t(h_{i,t}))$. Note that for each $t \in [T]$ and $v \in V$, we have $r_t(v) \geq \text{cost}(d_t; \texttt{OPT})/w_t(v)$. In particular, this holds for each $h_{i,t}$, showing that each F_i is non-empty. So let our answer $\texttt{ALG}$ consist of an arbitrary c_i from each F_i.

3-approximation. Fix a $t \in [T]$ and a $v \in C_{i,t}$. Then by properties of Plesník's algorithm listed above, and our construction of $\texttt{ALG}$, $d_t(v, \texttt{ALG}) \leq d_t(v, c_i) \leq d_t(v, h_{i,t}) + d_t(h_{i,t}, c_i) \leq 3r_t(v) \implies w_t(v) \cdot d_t(v, c_i) \leq 3\texttt{opt}_t$. The homogeneity of Ψ yields the desired approximation.

The algorithm runs in time $O(\varepsilon^{-O(T^2)}) \cdot O(k)^{O(kT)} \cdot \text{poly}(n) = O_{\varepsilon,k,T}(1) \cdot \text{poly}(n)$.

5 FPT Approximations in Well-structured Metrics

5.1 Metrics with Bounded Scatter Dimension

In this section, we study aggregate clustering in the case where all the T metrics have bounded *ε-scatter dimension*. This notion of metric dimension was originally introduced by [ABB+23] to obtain $O_{k,\varepsilon}(1) \cdot \text{poly}(n)$-time $(1+\varepsilon)$-approximation algorithms (ie, efficient parameterized approximation schemes, or EPASes for short) for a large class of center-based k-clustering problems, for example the $T = 1$ case of (k, z)-clustering.

Definition 31 (ε-scattering, ε-scatter dimension [ABB+23, Definition IV.1])**.** *An ε-scattering in a metric space $M = (C \cup F, d)$ is a sequence $(x_1, p_1), (x_2, p_2), \ldots, (x_l, p_l)$ s.t.*

- *$\forall i \in [l]$, $x_i \in F$ and $p_i \in C$;*
- *[covering] $\forall 1 \leq j < i \leq l$, $d(x_i, p_j) \leq 1$; and*
- *[ε-refutation] $\forall i \in [l]$, $d(x_i, p_i) > 1 + \varepsilon$.*

The ε-scatter dimension of a family of metric spaces $\mathcal{M}$ is the maximum length of any ε-scatter of an $M \in \mathcal{M}$.

A key component of the $(T = 1)$ EPASes of [ABB+23] is an EPAS for the special case of unweighted k-supplier. We recall their algorithm, following their presentation. At a high level, they maintain a k-sized solution $X \subseteq F$, and while this solution is not a $(1+\varepsilon)$-approximation, clients with high cost are identified to guide a recomputation of X; the bounded ε-scatter dimension is used to show a bound on the number of recomputations needed. In more detail, X is initialized arbitrarily as $X = \{x_1, \ldots, x_k\}$. Empty buckets $B_1, \ldots, B_k$ are also initialized, corresponding to clusters that would eventually be served by $x_1, \ldots, x_k$ respectively. We assume we know the cost $\texttt{opt}$ of the optimal solution[6]. While there exists a client violating the desired $(1+\varepsilon)$-approximation of $\texttt{opt}$—i.e. while $\exists v \in C : d(v, X) > (1+\varepsilon)\texttt{opt}$—the algorithm guesses the index $i \in [k]$ of the optimal cluster that serves v in $\texttt{OPT}$. The client v is then placed in the bucket B_i, and x_i is recomputed so that $\forall u \in B_i, d(u, x_i) \leq \texttt{opt}$. If such an x_i cannot be obtained, then the algorithm asserts failure and restarts from the initialization; but with positive probability $1/k$ we don't fail (because with probability $1/k$ we guess i correctly). Now here is the key insight, that each bucket B_i can't grow too large. Indeed, consider the sequence of requests $(v_i^{(1)}, \ldots, v_k^{(\ell)})$ added to B_i over time, and consider

[6]We can approximate it by bucketing by powers of $1+\varepsilon$ and guessing over the $O(\varepsilon^{-1} \log n)$ options.

the sequence of candidate locations for x_i over time, $(x_i^{(1)}, \ldots, x_i^{(\ell)})$. These two sequences form an ε-scatter (after we normalize distances so $\texttt{opt} = 1$). In particular, if the ε-scatter dimension of d is upper-bounded by some $\lambda(\varepsilon)$, then every bucket B_i only grows to size $\lambda(\varepsilon)$ until we either assert success or failure. As there there are k buckets overall, each iteration of the algorithm requires at most $k\lambda(\varepsilon)$ random choices each of which is correct with probability $1/k$; so with constant probability, we successfully find a solution after $O_{\varepsilon,k}(1)$ re-initializations.

For our aggregate clustering problem, we generalize the above. Rather than guessing a single partition of the clients into clusters served by $x_1, \ldots, x_k$, we need to guess T different partitions, one for each metric: we now have buckets $B_{i,t}$ for each $i \in [k], t \in [T]$, and we separately guess $\texttt{opt}_t = \text{cost}(d_t; \texttt{OPT})$ for each t. These metric-wise buckets allows us to mostly decouple the different metrics: when we find a violation in metric d_t, i.e. get $\text{cost}(d_t; X) > (1 + \Theta(\varepsilon))\texttt{opt}_t$, we select a random index $i \in [k]$ and add to bucket $B_{i,t}$, just as is done for the $T = 1$ case. Only when recomputing x_i do we take the other metrics into account; we choose x_i to obey the constraints of all the buckets $B_{i,t}$ across all $t \in [T]$, that is $\forall t \in [T] \forall u \in B_{i,t}, d_t(u, x_i) \leq \texttt{opt}_t$. Observe that each bucket $B_{i,t}$ still has bounded size, as it provides witness of an ε-scatter in metric d_t. To be precise, if one considers the sequence of requests $(v_i^{(1)}, \ldots, v_i^{(\ell)})$ in $B_{i,t}$ and the sequence of candidate positions for x_i *at the moment immediately before request* v_i^j *was added*[7], $(x_i^{(1)}, \ldots, x_i^{(\ell)})$, then these sequences form an ε-scatter for d_t. In particular, each iteration of the algorithm makes $kT\lambda(\varepsilon)$ random choices, and so it succeeds with constant probability after $O_{\varepsilon,k,T}(1)$ rounds.

Theorem 32. *When all T metrics have ε-scatter dimension at most $\lambda(\varepsilon)$, for any homogeneous aggregator Ψ there is an algorithm for Ψ-aggregate (unweighted) k-supplier runs in time $O_{\varepsilon,k,T}(1)\cdot\text{poly}(n)$, and produces a $(1+\varepsilon)$-approximation with constant probability.*

When $T = 1$, the k-supplier EPAS generalizes [ABB+23] to weighted k-center (by initializing with the help of a $O(1)$-approximation), then to weighted k-median (by a random sampling argument), and then to more general k-clustering problems. We adapt some of these ideas to the aggregate setting, and prove the following theorem.

Theorem 33. *When all T metrics have ε-scatter dimensions upper-bounded by some $\lambda(\varepsilon)$, for any homogeneous aggregator Ψ there is an algorithm for Ψ-aggregate k-median that runs in time $O_{\varepsilon,k,T}(1) \cdot \text{poly}(n)$, and produces $(1+\varepsilon)$-approximations with constant probability.*

Our algorithm for proving Theorem 33 is a direct extension of the analogous algorithm at $T = 1$ [ABB+23], so we recall the $T = 1$ unweighted k-supplier algorithm that we summarized from the literature [ABB+23] in Section 5.1. Let us understand how this is extended [ABB+23] to weighted k-supplier and weighted k-median at $T = 1$.

For the k-supplier problem with weights, X is initialized as Plesník's 3-approximation [Ple87], and labeled as $X = \{c_1, \ldots, c_k\}$. As in the unweighted case, empty buckets $B_1, \ldots, B_k$ are also initialized, corresponding to clusters that would eventually be served by $c_1, \ldots, c_k$ respectively. While there exists a client violating the desired $(1+\varepsilon)$-approximation—i.e. while $\exists v \in C : d(v, X) > (1+\varepsilon)\texttt{opt}/w(v)$, where $\texttt{opt}$ is a guess of the optimum—the algorithm guesses the index $i \in [k]$ of the optimal cluster that serves v in $\texttt{OPT}$. The client-radius pair $(v, \texttt{opt}/w(v))$ is then placed in the bucket B_i, and c_i is recomputed so that $\forall u \in B_i, d(u, c_i) \leq \texttt{opt}/w(u)$. If such a c_i cannot be obtained, then the algorithm asserts failure and restarts with a fresh 3-approximation; but this failure probability is $(1 - 1/k)$, i.e. the probability that i was guessed incorrectly. By construction of B_i, pairs in it with the same radius (say r) appear in an ε-scatter (after scaling distances down by r); and it can be shown [ABB+23] that there are only $O_\varepsilon(1/\varepsilon \cdot \log 1/\varepsilon)$ different radii appearing in B_i because we initialize X as a 3-approximation. So if the ε-scatter dimension is upper-bounded by some $\lambda(\varepsilon)$, then iterations involving B_i succeed with probability $(1/k)^{O(1/\varepsilon \cdot \log 1/\varepsilon)\cdot\lambda(\varepsilon)}$, leading to an overall success probability of $(1/k)^{O(1/\varepsilon\cdot\log 1/\varepsilon)\cdot k\lambda(\varepsilon)}$ each time we reinitialize.

The above approach for k-supplier is then generalized to other k-clustering problems, including k-median, by computing an upper bound $u(v)$ for each $d(v, \texttt{OPT})/w(v)$, and running the above algorithm with these $u(v)$'s in place of $\texttt{opt}/w(v)$. The crucial modification is that violation to the desired $(1+\varepsilon)$-approximation is

[7] We remark that, unlike the $T = 1$ case, the center x_i could take on more than ℓ candidate positions, because the the position of x_i could be changed due to a request in any bucket $B_{i,t'}$ for $t' \in [T]$; nevertheless, even if x_i is changed due to a different bucket $B_{i,t'}$ we still guarantee that x_i covers requests in $B_{i,t}$, and so one can construct an ε-scatter.

no longer clientwise—as $\texttt{opt}$ is no longer a single client's cost, a violation may take a form like $\sum_{v \in C} d(v, X) > (1+\varepsilon)\texttt{opt}$. Nevertheless, once a cost violation is found, a clever sampling [ABB$^+$23, Algorithm 1] is possible to pick a client v that is indeed a violation on its own, i.e. $d(v, X) > (1+\varepsilon')d(v, \texttt{OPT})$ for some $\varepsilon' = \Theta(\varepsilon)$. Then $(v, (1+\varepsilon')d(v, X))$ is added to bucket B_i for a guessed $i \in [k]$. The sampling also ensures that $u(v) \geq d(v, \texttt{OPT})$ is small for a sampled v, so that smaller radii are more likely to be added to a bucket. Once a pair (v, r) is added to a bucket, any subsequent (v, r') in the same bucket must satisfy $r' < r$; so ensuring smaller radii upper-bounds the length of a bucket, as desired.

We now extend the above to $T > 1$, proving Theorem 33 (restated below). We keep our analysis brief, focusing on the differences from the $T = 1$ case [ABB$^+$23].

Proof of Theorem 33. As before, we assume via bucketing that there are $O(\varepsilon^{-1} \log n)$ distances per metric, allowing us to guess $\texttt{opt}_t = \text{cost}(d_t; \texttt{OPT})$ for each $t \in [T]$ within the desired running time. We also compute upper bounds $u_t(v)$ for every $d_t(v, \texttt{OPT})/w_t(v)$ as follows: if $z = \infty$ then $u_t(v) = \texttt{opt}_t/w_t(v)$; otherwise, $u_t(v) = 2 \cdot \min\{r : w_t(B_t(v, r)) \geq 3\texttt{opt}_t/r\}$. Using these $u_t(v)$'s as our radii, we run our polytime 3-approximation algorithm from Section 3.1; that is, we obtain $X = \{c_1, \ldots, c_k\} : \forall t \in [T], v \in C, d_t(v, X) \leq 3u_t(v)$.

Observe that, by homogeneity of Ψ, if we had $\text{cost}(d_t; X) \leq (1+\varepsilon)\texttt{opt}_t$ for every $t \in [T]$, then we would be done. So there must be some t violating this inequality, i.e. such that $\text{cost}(d_t; X) > (1+\varepsilon)\texttt{opt}_t$; and we can identify this t in $\text{poly}(n, T)$ time by simply computing all T costs. While such a t exists, we run a *violation loop* to improve the solution X. An iteration of this loop begins with an attempt to find a $v \in C$ with the following properties w.r.t. d_t and X:

- (witness) $d_t(v, X) > (1+\varepsilon/3)d_t(v, \texttt{OPT})$, and
- (upper bound) $u_t(v) = O(kd_t(v, X)/\varepsilon)$.

For k-supplier, this v is easy to find, as it was in the $T = 1$ case: $\text{cost}(d_t; X)$ must equal some $w_t(v) \cdot d_t(v, X)$, so we pick that v. Then $d_t(v, X) = \text{cost}(d_t; X) > (1+\varepsilon)\texttt{opt}_t \geq (1+\varepsilon)d_t(v, \texttt{OPT})$, satisfying the witness condition. Also $u_t(v) \cdot w_t(v) = \texttt{opt}_t \leq \text{cost}_t(d_t; X)/(1+\varepsilon) = w_t(v) \cdot d_t(v, X)/(1+\varepsilon)$, implying $u_t(v) \leq d_t(v, X)/\varepsilon$, which satisfies the upper bound condition.

When $z \in \mathbb{N}$, we observe the following, again in line with the $T = 1$ case: to satisfy the witness condition alone, sampling with probability $\propto w_t(v) \cdot d_t(v, X)$ from C suffices via an averaging argument. But to satisfy both conditions, we instead sample from a smaller client-set A_t in which the upper bound condition already holds. The key technical result, like in the $T = 1$ case, is that "many witnesses lie in A_t" or, more precisely, that a large fraction of the weighted cost of A_t is contributed by witnesses. This result involves the exact same construction and analysis as the analogous result [ABB$^+$23, Lemma V.10] in the $T = 1$ case. Where that proof constructed desirable "heavy witness sets" $W_1^+, \ldots, W_k^+ \subseteq A$, we now have for each $t \in [T]$, analogous sets $W_{1,t}^+, \ldots, W_{k,t}^+ \subseteq A_t$, satisfying the same guarantees within that d_t. These constructions in the T different metrics do not interact with one another, so we can analyze each one exactly as for the $T = 1$ analysis. Because of this direct similarity, we omit the details of this proof, and only state the extended result below.

Lemma 34 (Generalization of [ABB$^+$23, Lemma 5.10] to $T > 1$)**.** *Fix a violating metric, i.e. a $t \in [T]$: $\text{cost}(d_t; X) > (1+\varepsilon)\texttt{opt}_t$. Let v be a client sampled from the set $A_t := \left\{v \in C : d_t(v, X) = \Omega\left(\frac{\varepsilon \cdot u_t(v)}{k}\right)\right\}$, with probability proportional to $w_t(v) \cdot d_t(v, X)$. Then with probability $\Omega(\varepsilon)$, v satisfies the witness condition and the upper bound condition.*

Having sampled this v with the desired properties, we guess its "correct cluster" $i \in [k]$, i.e. where the unknown optimal centers are $o_1, \ldots, o_k$, we guess that $d_t(v, \texttt{OPT}) = d_t(v, o_i)$. We then add $(v, r_t(v))$ to the bucket $B_{i,t}$; here $r_t(v) = u_t(v)$ for k-supplier, and $r_t(v) = d_t(v, X)/(1+\varepsilon/3)$ otherwise. So $r_t(v) < d_t(v, X)$. Then, we attempt to recompute c_i such that

$$\forall t' \in [T], p \in B_{i,t'}, d_{t'}(p, c_i) \leq r_{t'}(p).$$

This is the *only major step where multiple metrics are involved*; it still holds, however, that if each iteration guesses the correct i then this c_i exists. If such a c_i does not exist, then we assert a FAIL state, and restart

with a fresh initialization of X; otherwise we continue looping by looking for another violation, which might occur at a different $t' \in [T]$.

To establish correctness, we must bound the failure probability of this algorithm: note that each guess of $i \in [k]$ succeeds with probability $(1/k)$, so for iterations that augment a particular bucket $B_{i,t}$, we succeed with probability $(1/k)^{|B_{i,t}|}$. To bound this bucket size, observe that for every radius r that appears in $B_{i,t}$, the set $B_{i,t}|_r := \{(v, r_v) : r_v = r\}$ is an ε-scatter in d_t (when scaled down by r). That is, $\left|B_{i,t}|_r\right| \leq \lambda(\varepsilon)$. So we must bound the number of different radii appearing in $B_{i,t}$. Since the bucket has a fixed t, this follows exactly as for the $T = 1$ case [ABB$^+$23, Section II, p. 1384], yielding $O(1/\varepsilon \cdot \log 1/\varepsilon)$ different radii in $B_{i,t}$. So over the kT buckets, we obtain a success probability of $(1/k)^{O_\varepsilon(1)\cdot kT}$, meaning that we have succeeded with constant probability after $O_{k,\varepsilon,T}(1)$ repetitions. □

Remark 35 (Barrier at algorithmic scatter dimension)**.** *The authors of [ABB$^+$23] also consider a weaker notion of* algorithmic ε-scatter dimension *(where, roughly speaking, the length of ε-scatters are only bounded if the sequence of centers x_i is chosen according to a certain algorithm), and their EPASes extend to metrics of bounded algorithmic scatter dimension. For example in the case of k-supplier, when recomputing the center c_i for some bucket B_i, they just use the algorithm for computing the center rather than choosing x_i to be an arbitrary point that covers B_i. We are aware of two follow-up works [GI25, BGI25] that build on the framework of [ABB$^+$23], both of which apply also for algorithmic scatter dimension. In contrast, our aggregate algorithm* does not extend *to the more general setting; this is because we recompute x_i in a way that covers all the buckets $B_{i,t}$ across $t \in [T]$ (and such an x_i is guaranteed to exist, if we guessed the buckets correctly), whereas an algorithmic version would require that x_i was chosen to simultaneously satisfy the algorithms of all T metrics (and such an x_i does* not *necessarily exist).*

5.2 Bounded-treewidth Graphs

Theorem 36. *When the base graph G has treewidth* tw *and the aggregator $\Psi = sum$, then:*

- *there is an exact algorithm for sum-aggregate (k, z)-clustering that runs in time $n^{O(T\cdot \mathsf{tw})}$.*
- *for any $\varepsilon > 0$, there is a $(1+\varepsilon)$-approximation algorithm for sum-aggregate (k, z)-clustering that runs in time $(\frac{\log n}{\varepsilon})^{O(T\cdot \mathsf{tw})} \cdot \mathrm{poly}(n) = O_{\varepsilon,T,\mathsf{tw}}(1) \cdot \mathrm{poly}(n)$.*

In this section we focus on the k-median case; the ideas extend naturally to (k, z)-clustering and k-supplier. We prove Theorem 36 by adapting a folklore dynamic program for solving k-median on bounded-treewidth graphs. (Recall that a tree decomposition of a graph G is a tree $\mathcal{T}$ of *bags*, where each bag is a subset of vertices $V(G)$, that satisfies two properties: (1) for every edge (u, v) in G, there is a bag containing both u and v, and (2) for every vertex v in $V(G)$, the bags containing v induce a connected subtree of $\mathcal{T}$. The *width* of the tree decomposition is 1 less than the maximum bag size, and the *treewidth* of a graph is the minimum width of any tree decoposition. A tree has treewidth 1.) To the best of our knowledge, the DP for k-median has not been written down explicitly in the setting of bounded treewidth graphs, but the technique is standard. Our presentation is based on a DP that appeared [ARR98, KR07, CFS21] as part of a PTAS for k-median in Euclidean metrics. In this section we explain the main idea of the algorithm and defer the formal details to Appendix 5.2.

We begin by reviewing the folklore DP. Every bag B of the tree decomposition $\mathcal{T}$ acts as a small separator between an "inside" piece (induced by the vertices in the subtree of $\mathcal{T}$ rooted at B) and an "outside" piece (induced by all vertices outside the subtree rooted at B). For each separator, the DP guesses the *interface* of the optimal solution $\mathtt{OPT}$ between the two pieces. In the case of k-median, the interface of bag $B = \{v_1, \ldots, v_{\mathsf{tw}}\}$ consists of the distance of every v_i to the closest open center in the inside (resp. outside) piece, as well as the number of centers opened in the inside (resp. outside) piece in the optimal solution. Note that there are $n^{O(\mathsf{tw})}$ possible interfaces for any bag, as there are $O(n^2)$ possible distances for each v_i; by rounding distances to the nearest power of $(1 + \frac{\varepsilon}{\log n})$, we could reduce the number of interfaces to $(\frac{\log n}{\varepsilon})^{O(\mathsf{tw})}$.

The DP table contains a cell for every bag B and every possible interface of B, where the value of the cell represents the cost paid by vertices in the inside piece under the best solution that respects the interface. The value of a DP cell at bag B can be computed from the DP cells of the children of B, by taking a minimum over all combinations of children interfaces which are *consistent* with the interface of B (checking

the consistency of interfaces can be done in a straightforward manner in polynomial time [ARR98, CFS21]). The overall runtime is $n^{O(\mathsf{tw})}$, or $(\frac{\log n}{\varepsilon})^{O(\mathsf{tw})} \cdot \mathrm{poly}(n)$ if we tolerate $1+\varepsilon$ approximation.

To adapt the DP to the aggregate setting, we simply change the notion of *interface*: rather than guessing just one interface, we guess a different interface for each of the T metrics $d_1, \ldots, d_T$. The cells of our DP table now consist of a bag $B = \{v_1, \ldots, v_{\mathsf{tw}}\}$ and a set of T tuples; the t-th tuple consists of the distance *with respect to* d_t of every v_i to the closest (with respect to d_t) open center in $\mathtt{OPT}$ in each piece, as well as the cost paid by each piece in the d_t metric under $\mathtt{OPT}$. (Note: the reason that we need to guess a different interface for each metric is that, even though the optimal set of centers $\mathtt{OPT}$ for the aggregate problem stays consistent throughout the T metrics, a vertex v could be served by a different center of $\mathtt{OPT}$ in each different metric d_t.) There are $n^{O(T\cdot\mathsf{tw})}$ possible interfaces. The DP can be computed in a bottom-up fashion, as in the non-aggregate setting, for a total runtime of $n^{O(T\cdot\mathsf{tw})}$. If one allows $1+\varepsilon$ approximation, the runtime improves to $(\frac{\log n}{\varepsilon})^{O(T\cdot\mathsf{tw})} \cdot \mathrm{poly}(n)$, which is at most $\varepsilon^{-O(T^2\cdot\mathsf{tw}^2)} \cdot \mathrm{poly}(n) = O_{\varepsilon,T,\mathsf{tw}}(1) \cdot \mathrm{poly}(n)$ (see Fact 39).

Remark 37. *Observe that if* $\mathsf{tw} = o\left(\frac{\log n}{\log\log n}\right)$*, then Theorem 36 provides a* $(1+\varepsilon)$*-approximation algorithm in* $2^{o(\log n)\cdot T\cdot\log(1/\varepsilon)} \cdot \mathrm{poly}(n) = O_{\varepsilon,T}(1) \cdot \mathrm{poly}(n)$ *time (by Fact 39).*

Proof of Theorem 36

We describe in detail the exact $n^{O(T\cdot\mathsf{tw})}$ algorithm for stochastic aggregate k-median. At the end, we sketch the changes needed for the $1+\varepsilon$ approximation, and the changes needed to extend the result to general Ψ-aggregate (k,z)-clustering and k-center as claimed in the theorem. Let $F \subseteq V(G)$ denote the facilities, and for every $t \in [T]$ let $w_t : V(G) \to \mathbb{R}_{>0}$ denote the weights of the clients in the t-th metric. We let $C \subseteq V(G)$ denote the set of *non-trivial clients*, i.e. those clients c with $w_t(v) > 0$ for at least one metric t.

Let $\mathcal{T}$ be a rooted tree decomposition for G, where $\mathcal{T}$ has width $O(\mathsf{tw})$, depth $O(\log n)$, and is a binary tree. Such a $\mathcal{T}$ can be found in time $2^{O(\mathsf{tw})} \cdot \mathrm{poly}(n)$ [BH98]. Moreover, we assume without loss of generality (as in [ABM+19]) that facilities F and the non-trivial clients C appear *only* in leaf bags of $\mathcal{T}$, and that each facility and non-trivial client appears in exactly one bag. This assumption can be enforced WLOG by *duplicating* each client or facility v to make a copy v' connected to v by 0-weight edge, and updating the tree decomposition to contain a leaf bag $\{v, v'\}$; the vertex v is treated as a Steiner vertex (ie., v is treated if it was not a facility or client, meaning $v \notin F$ and $w_t(v) = 0$ for all t), and the vertex v' (which appears in exactly one bag in $\mathcal{T}$, in a leaf bag) is treated as the client/facility. This duplication process could increase the arity of the tree decomposition $\mathcal{T}$, but further duplication can reduce the arity back to 2 while only increasing the number of vertices in G (and the depth of $\mathcal{T}$) by a constant factor.

For each $t \in [T]$, let R_t be the set of possible pairwise distances under the metric d_t. Note that $|R_t| = O(n^2)$. For any node b in $\mathcal{T}$, let $X_b \subseteq V$ be the bag of vertices associated with the node. Let $\mathcal{T}_b$ be the subtree rooted at b, and let $V_b \subseteq V$ be the vertices appearing in bags of $\mathcal{T}_b$.

Definition of DP. Our DP entries are defined over the following:

- A node b in the tree decomposition $\mathcal{T}$.
- A budget $k' \in \mathbb{N}$, representing the number of centers we can open in $\mathcal{T}_b$.
- An "outer distance" vector $\boldsymbol{o}$, which records a distance $o_{t,x} \in R_t$ for every vertex $x \in X_b$ and every metric $t \in [T]$. (This distance represents an assumption that vertex $x \in X_b$ is within distance $o_{t,x}$ of some open center outside of the subtree $\mathcal{T}_B$.)
- An "inner distance" vector $\boldsymbol{i}$, which records a distance $i_{t,x} \in R_t$ for every vertex $x \in X_b$ and every metric $t \in [T]$. (This distance represents a requirement that we open some center in $\mathcal{T}_b$ within distance $i_{t,x}$ of vertex $x \in X_b$.)

The DP entry $\mathrm{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}]$ represents the minimum cost paid by clients in $\mathcal{T}_b$ if we open k' centers in $\mathcal{T}_b$, while making the assumption that we have already opened facilities outside $\mathcal{T}_b$ as specified by the interface $\boldsymbol{o}$ and while obeying the restriction that we must open centers in $\mathcal{T}_b$ to satisfy $\boldsymbol{i}$. Formally,

$$\mathrm{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}] \coloneqq \min_{\substack{S \subseteq F \cap V_b : |S| = k' \\ \forall t \in [T], \forall x \in X, d_t(x,S) = i_{t,x}}} \mathrm{cost}(V_b, S, \boldsymbol{o}) \tag{1}$$

where

$$\text{cost}(b, S, \boldsymbol{o}) \coloneqq \sum_{t\in[T]} \sum_{c\in V_b} w_t(c) \cdot \min\left(d_t(c,S), \min_{x\in X_b}(d_t(c,x) + o_{t,x})\right). \tag{2}$$

If no such S exists then we set $\text{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}] = \infty$.

Objective. Observe that if b is the root bag of the tree decomposition, then the cost of the optimal aggregate solution $\texttt{OPT}$ is precisely $\min_{\boldsymbol{i}} \text{DP}[b, k, \langle\infty, \ldots, \infty\rangle, \boldsymbol{i}]$.

Base Cases. Our base case is when b is a leaf, and in this case we simply evaluate the DP definition by enumeration. We guess the k' centers $S \subseteq X_b$ that are open; if there is no guess that satisfies the $\boldsymbol{i}$ constraint then return ∞, otherwise we find the S that minimizes the cost paid by clients in X_b. This can be done in $2^{O(\mathsf{tw})} \cdot \text{poly}(n, T)$ time.

Recurrence. Let the two children of b be ℓ and r. We recurse on ℓ and r with distance vectors that are *consistent* per the following definition.

Definition 38. *Let b node of $\mathcal{T}$ with children l and r. Let $(k', \boldsymbol{o}, \boldsymbol{i})$ be a tuple that defines a DP entry associated with b, consisting of an inner vector of distances, an outer vector of distances, and a budget k'. Let $(k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell})$ and $(k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r})$ be tuples defining DP entries associated with ℓ and r respectively. We say that $(k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r})$ is consistent with $(\boldsymbol{o}, \boldsymbol{i}, \boldsymbol{c}, k')$ if the following conditions all hold.*

- $k'_l + k'_r = k'$.
- [Constraint $\boldsymbol{i}$ is satisfied.] *For every metric $t \in [T]$ and vertex $x \in X_b$, there exists some $y \in X_\ell$ (or some $y \in X_r$) such that $i_{t,x} = d_t(x, y) + i^{\ell}_{t,t}$ (or, respectively, $i_{t,x} = d_t(x, t) + i^{r}_{t,t}$). In other words, the restriction $\boldsymbol{i}$ for bag b is satisfied by a restriction that either $\boldsymbol{i}^{\ell}$ or $\boldsymbol{i}^{r}$ imposes on a child bag.*
- [Assumptions $\boldsymbol{o}^{\ell}, \boldsymbol{o}^{r}$ are justified.] *For each $t \in [T]$ and $y \in X_\ell$ (and similarly for $y \in X_r$), there some $x \in X_b$ such that either $o^{\ell}_{t,y} = d_t(y, x) + o_{t,x}$ or $o^{\ell}_{t,y} = d_t(y, x) + i^{r}_{t,y}$. In other words, the assumption $\boldsymbol{o}^{r}$ on bag r is justified either by the assumption $\boldsymbol{o}$ on the parent bag b or by the constraint $\boldsymbol{i}$ on the sibling bag ℓ.*

To evaluate $\text{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}]$, we guess over all $n^{O(T\cdot\mathsf{tw})}$ possibilities for $\boldsymbol{o}^{\ell}, \boldsymbol{o}^{r}, \boldsymbol{i}^{\ell}, \boldsymbol{i}^{r}, k'^{\ell}, k'^{r}$ and verify consistency in $\text{poly}(n)$ time per guess; we return the minimum sum of consistent child DP entries. In notation:

$$\text{DP}[b, k'\boldsymbol{o}, \boldsymbol{i}] \leftarrow \min_{\substack{(k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r}) \text{ consistent} \\ \text{with } (k', \boldsymbol{o}, \boldsymbol{i})}} \text{DP}[\ell, k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}] + \text{DP}[r, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r}]. \tag{3}$$

Since there are $n^{O(T\cdot\mathsf{tw})}$ entries, filling the DP table bottom-up leads to a total running time of $n^{O(T\cdot\mathsf{tw})}$.

Correctness of recurrence. We show that recurrence (3) correctly outputs the value of the DP cell according to the definition (1).

First we show that (3) $\leq$ (1), by observing that for any $\text{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}]$, there is a choice of consistent $(k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r})$ such that $\text{DP}[\ell, k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}] + \text{DP}[r, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r}] = \text{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}]$. Indeed, let S be the set of centers minimizing (1), such that $\text{cost}(b, S, \boldsymbol{o}) = \text{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}]$. Let ℓ and r be the children of bag b, and let $S_\ell = V_\ell \cap S$ and $S_r = V_r \cap S$. We define variables as follows:

- Set $k'^{\ell} \coloneqq |S_\ell|$ and $k'^{r} \coloneqq |S_r|$.
- For every $t \in [T]$ and $y \in X_\ell$ (and similarly for $y \in X_r$), set $i^{\ell}_{t,y} \coloneqq d_t(y, S_\ell)$,
- and set $o^{\ell}_{t,y} \coloneqq \min\left(d_t(y, S_r), \min_{x\in X_b}(d_t(y, x) + o_x)\right)$.

As $S = S_\ell \cup S_r$, it is immediate that $\text{cost}(b, S, \boldsymbol{o}) = \text{cost}(\ell, S_\ell, \boldsymbol{o}^{\ell}) + \text{cost}(r, S_r, \boldsymbol{o}^{r})$, meaning that $\text{DP}[\ell, k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}] + \text{DP}[r, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r}] = \text{DP}[b, k', \boldsymbol{o}, \boldsymbol{i}]$. It remains to check the consistency of the variables. Consistency follows from properties of tree decomposition: the [Constraint $\boldsymbol{i}$ is satisfied] property follows from the fact that any shortest path from $x \in X_b$ to S_ℓ (resp. S_r) passes through some vertex of X_ℓ (resp. S_r), and the

[Assumptions $\boldsymbol{o}^\ell, \boldsymbol{o}^r$ are justified] property follows from the fact that any shortest path between V_ℓ to V_r passes through some vertex of X_b.

We now need to show the other direction, (1) $\leq$ (3): that is, every choice of a consistent tuple $(k'^\ell, \boldsymbol{o}^\ell, \boldsymbol{i}^\ell, k'^r, \boldsymbol{o}^r, \boldsymbol{i}^r)$ corresponds to some candidate set of centers $S \subseteq V_b$ such that $\text{cost}(b, S, \boldsymbol{o}) = \text{DP}[\ell, k'^\ell, \boldsymbol{o}^\ell, \boldsymbol{i}^\ell] + \text{DP}[r, k'^r, \boldsymbol{o}^r, \boldsymbol{i}^r]$. Indeed, we can choose $S \coloneqq S_\ell \cup S_r$, where S_ℓ (resp. S_r) is the set of centers minimizing $\text{DP}[\ell, k'^\ell, \boldsymbol{o}^\ell, \boldsymbol{i}^\ell]$ (resp. $\text{DP}[r, k'^r, \boldsymbol{o}^r, \boldsymbol{i}^r]$). The set S is a valid candidate set of centers: the size of S is $k'^\ell + k'^r = k'$, and (by properties of tree decomposition and the [Constraint $\boldsymbol{i}$ is satisfied] condition of consistency) for every $t \in [T]$ and $x \in X_b$ we have $d_t(x, S) = i_{t,x}$. Moreover, we have that $\text{cost}(b, S, \boldsymbol{o}) = \text{cost}(\ell, S_\ell, \boldsymbol{o}^\ell) + \text{cost}(r, S_r, \boldsymbol{o}^r)$; the properties of tree decomposition and the [Assumptions $\boldsymbol{o}^\ell, \boldsymbol{o}^r$ are justified] property of consistency implies that for every vertex $c \in V_\ell$ (resp. V_r), the cost paid by c under $(S, \boldsymbol{o})$ is the same as the cost paid by c under $(S_\ell, \boldsymbol{o}^\ell)$ (resp. $(S_r, \boldsymbol{o}^r)$).

$1+\varepsilon$ approximation in FPT time. We have described an exact algorithm that runs in time $n^{O(T \cdot \mathsf{tw})}$. We now sketch how to improve the running time, if one allows $1+\varepsilon$ approximation. By a standard reduction, we can assume that that aspect ratio (ie, the ratio of the largest distance to smallest distance) in each metric is $\text{poly}(n, \varepsilon^{-1})$ at the cost of $1+\varepsilon$ approximation. Note that in the exact algorithm, we duplicated some vertices and connected them by 0-weight edges, to enforce the assumption that the facilities and clients appear only in leaf bags — now that we want bounded aspect ratio, we instead add $\frac{\varepsilon}{\text{poly}(n)}$-weight edges, again losing only a $1+\varepsilon$ factor in the approximation. We then round every distance in each metric to the closest power $(1 + \frac{\varepsilon}{\log n})$; now there are $O(\log_{1+\frac{\varepsilon}{\log n}} n) = O(\varepsilon^{-1} \log^2 n)$ distinct distances rather than $O(n^2)$. In our DP, rather than guessing $o_{t,x}$ and $i_{t,x}$ from the set of $O(n^2)$ distances, we guess the distances only to the closest power of $(1 + \frac{\varepsilon}{\log n})$. When checking consistency in Definition 38, we modify the conditions [Constraint $\boldsymbol{i}$ is satisfied] and [Assumptions $\boldsymbol{o}^\ell, \boldsymbol{o}^r$ are justified] to only require that the equalities hold up to a $1 + \frac{\varepsilon}{\log n}$ approximation factor. The overall running time is $(\frac{\log n}{\varepsilon})^{O(T \cdot \mathsf{tw})} \text{poly}(n)$, which is bounded above by $\varepsilon^{-O(T^2 \cdot \mathsf{tw}^2)} \text{poly}(n)$ by Fact 39.

Each recursive call of the DP accumulates $1 + \frac{\varepsilon}{\log n}$ error due to the rounding. To be more precise, if the recursive evaluation of the DP cell at bag b has value κ, and the bag b has height h in the tree $\mathcal{T}$, then one can show (by an inductive argument) that there is a valid set of centers $S \subseteq V_b$ with cost at most $(1 + \frac{\varepsilon}{\log n})^h \cdot \kappa$ according to the definition (1). As the height of the tree decomposition $\mathcal{T}$ is $O(\log n)$, overall we get a solution with approximation ratio $1 + O(\varepsilon)$. Rescaling ε by a constant factor yields the claim.

Generalizing to (k, z)-clustering and k-supplier. We can generalize the DP to work with stochastic aggregate (k, z)-clustering for any z. In the definition of the DP, we now use

$$\text{cost}(b, S, \boldsymbol{o}) \coloneqq \sum_{t \in [T]} \left(\sum_{c \in V_b} w_t(c) \cdot \min(d_t(c, S), \min_{x \in X_b} (d_t(c, x) + o_{t,x}))^z \right)^{1/z}$$

so that the DP entry at a bag b gives us the (k, z)-clustering cost of the subtree rather than k-median cost. We modify the recurrence: rather than just taking a sum of the two children DP entries, we set

$$\text{DP}[b, k'\boldsymbol{o}, \boldsymbol{i}] \leftarrow \min_{\substack{(k'^\ell, \boldsymbol{o}^\ell, \boldsymbol{i}^\ell, k'^r, \boldsymbol{o}^r, \boldsymbol{i}^r) \\ \text{consistent with} \\ (k', \boldsymbol{o}, \boldsymbol{i})}} \left(\left(\text{DP}[\ell, k'^\ell, \boldsymbol{o}^\ell, \boldsymbol{i}^\ell] \right)^z + \left(\text{DP}[r, k'^r, \boldsymbol{o}^r, \boldsymbol{i}^r] \right)^z \right)^{1/z}.$$

We use this recurrence because in (k, z)-clustering we have the property that $\text{cost}(b, S, \boldsymbol{o}) = \left(\text{cost}(\ell, S_\ell, \boldsymbol{o}^\ell)^z + \text{cost}(r, S_r, \boldsymbol{o}^r)^z \right)^1$ rather than the property that $\text{cost}(b, S, \boldsymbol{o}) = \text{cost}(\ell, S_\ell, \boldsymbol{o}^\ell) + \text{cost}(r, S_r, \boldsymbol{o}^r)$ that we used for k-median. The proof of correctness of the (k, z)-clustering recurrence is similar to that of the k-median recurrence, once we replace the linearity of k-median cost with the new property just described. Our $1+\varepsilon$ approximate DP also works for (k, z)-clustering, because distorting distances by a factor $1+\varepsilon$ only changes $\text{cost}(b, S, \boldsymbol{o})$ by at most a factor $1 + \varepsilon$.

By setting $z = \infty$, we recover the aggregate k-supplier problem. To be precise, our definition of the DP becomes

$$\text{cost}(b, S, \boldsymbol{o}) \coloneqq \sum_{t \in [T]} \max_{c \in V_b} \left(w_t(c) \cdot \min(d_t(c, S), \min_{x \in X_b} (d_t(c, x) + o_{t,x})) \right)$$

and

$$\mathrm{DP}[b, k'\boldsymbol{o}, \boldsymbol{i}] \leftarrow \min_{\substack{(k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r}) \text{ consistent} \\ \text{with } (k', \boldsymbol{o}, \boldsymbol{i})}} \max\left(\mathrm{DP}[\ell, k'^{\ell}, \boldsymbol{o}^{\ell}, \boldsymbol{i}^{\ell}], \mathrm{DP}[r, k'^{r}, \boldsymbol{o}^{r}, \boldsymbol{i}^{r}]\right).$$

Correctness follows from the fact that $\mathrm{cost}(b, S, \boldsymbol{o}) = \max(\mathrm{cost}(\ell, S_\ell, \boldsymbol{o}^{\ell}), \mathrm{cost}(r, S_r, \boldsymbol{o}^{r}))$.

References

[ABB+23] Fateme Abbasi, Sandip Banerjee, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. Parameterized approximation schemes for clustering with general norm objectives. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1377–1399. IEEE, 2023.

[ABM+19] Marek Adamczyk, Jarosław Byrka, Jan Marcinkowski, Syed M Meesum, and Michał Włodarczyk. Constant-factor FPT approximation for capacitated k-median. In *27th Annual European Symposium on Algorithms (ESA 2019)*, pages 1–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019.

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

[AFGW10] Ittai Abraham, Amos Fiat, Andrew V Goldberg, and Renato F Werneck. Highway dimension, shortest paths, and provably efficient algorithms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 782–793. SIAM, 2010.

[AGGN08] Barbara M Anthony, Vineet Goyal, Anupam Gupta, and Viswanath Nagarajan. A plant location guide for the unsure. In *the Symposium on Discrete Algorithms (SODA)*, 2008.

[AGGN10] Barbara Anthony, Vineet Goyal, Anupam Gupta, and Viswanath Nagarajan. A plant location guide for the unsure: Approximation algorithms for min-max location problems. *Mathematics of Operations Research*, 35(1):79–101, 2010. Preliminary version appeared in SODA 2008 [AGGN08].

[AGK+04] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing (SICOMP)*, 33(3):544–562, 2004.

[AISX24] Akanksha Agrawal, Tanmay Inamdar, Saket Saurabh, and Jie Xue. Clustering what matters: Optimal approximation for clustering with outliers. *J. Artif. Int. Res.*, 78, 2024.

[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.

[BCF25] Sayan Bhattacharya, Martín Costa, and Ermiya Farokhnejad. Fully dynamic k-median with near-optimal update time and recourse. In *the Symposium on Theory of Computing (STOC)*, pages 1166–1177, 2025.

[BCLP23] Sayan Bhattacharya, Martín Costa, Silvio Lattanzi, and Nikos Parotsidis. Fully dynamic k-clustering in $\tilde{O}(k$ update time. *Adv. in Neural Infomation Processing Systems (NeurIPS)*, 36:18633–18658, 2023.

[BCLP25] Sayan Bhattacharya, Martín Costa, Silvio Lattanzi, and Nikos Parotsidis. Fully dynamic k-center clustering made simple. In *the International Conference on Machine Learning (ICML)*, 2025.

[BEF+23] MohammadHossein Bateni, Hossein Esfandiari, Hendrik Fichtenberger, Monika Henzinger, Rajesh Jayaram, Vahab Mirrokni, and Andreas Wiese. Optimal fully dynamic k-center clustering for adaptive and oblivious adversaries. In *the Symposium on Discrete Algorithms (SODA)*, pages 2677–2727, 2023.

[BGI25] Sujoy Bhore, Ameet Gadekar, and Tanmay Inamdar. Coreset strikes back: Improved parameterized approximation schemes for (constrained) k-median/means. *arXiv preprint arXiv:2504.06980*, 2025.

[BH98] Hans L Bodlaender and Torben Hagerup. Parallel algorithms with optimal speedup for bounded treewidth. *SIAM Journal on Computing*, 27(6):1725–1746, 1998.

[BP25] Romain Bourneuf and Marcin Pilipczuk. Bounding ε-scatter dimension via metric sparsity. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3155–3171. SIAM, 2025.

[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 *the Symposium on Discrete Algorithms (SODA)*, pages 737–756, 2014.

[CAGK+19] Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. Tight fpt approximations for k-median and k-means. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132, pages 42–1. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019.

[CAHP+19] Vincent Cohen-Addad, Niklas Oskar D Hjuler, Nikos Parotsidis, David Saulpic, and Chris Schwiegelshohn. Fully dynamic consistent facility location. *Adv. in Neural Infomation Processing Systems (NeurIPS)*, 32, 2019.

[Car11] Constantin Carathéodory. Über den variabilitätsbereich der fourier'schen konstanten von positiven harmonischen funktionen. *Rendiconti Del Circolo Matematico di Palermo (1884-1940)*, 32(1):193–217, 1911.

[CC16] Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. *Journal of the ACM (JACM)*, 63(5):1–65, 2016.

[CCS24] Deeparnab Chakrabarty, Luc Cote, and Ankita Sarkar. Fault-tolerant k-supplier with outliers. In *the Symposium on Theoretical Aspects of Computer Science (STACS)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024.

[CFK+15] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*. Springer, 2015.

[CFS21] Vincent Cohen-Addad, Andreas Emil Feldmann, and David Saulpic. Near-linear time approximation schemes for clustering in doubling metrics. *Journal of the ACM (JACM)*, 68(6):1–34, 2021.

[CGK20] Deeparnab Chakrabarty, Prachi Goyal, and Ravishankar Krishnaswamy. The Non-Uniform k-Center Problem. *the ACM Transactions on Algorithms (TALG)*, 2020. Preliminary version in ICALP 2016.

[CGS18] TH Hubert Chan, Arnaud Guerqin, and Mauro Sozio. Fully dynamic k-center clustering. In *the International World Wide Web Conference*, pages 579–587, 2018.

[CGTS99] Moses Charikar, Sudipto Guha, Éva Tardos, and David B Shmoys. A constant-factor approximation algorithm for the k-median problem. In *the Symposium on Theory of Computing (STOC)*, 1999.

[CGTS02] Moses Charikar, Sudipto Guha, Éva Tardos, and David B Shmoys. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65(1):129–149, 2002. Preliminary version appeared in STOC 1999 [CGTS99].

[Chu15] Julia Chuzhoy. Excluded grid theorem: Improved and simplified. In *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*, pages 645–654, 2015.

[Chu16] Julia Chuzhoy. Improved bounds for the excluded grid theorem. *arXiv preprint arXiv:1602.02629*, 2016.

[CKMN01] Moses Charikar, Samir Khuller, David M Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *the Symposium on Discrete Algorithms (SODA)*, volume 1, pages 642–651. Citeseer, 2001.

[CL12] Moses Charikar and Shi Li. A dependent lp-rounding approach for the k-median problem. In *International Colloquium on Automata, Languages, and Programming*, pages 194–205. Springer, 2012.

[CMV22] Eden Chlamtáč, Yury Makarychev, and Ali Vakilian. Approximating fair clustering with cascaded norm objectives. In *the Symposium on Discrete Algorithms (SODA)*, 2022.

[CN19] Deeparnab Chakrabarty and Maryam Negahbani. Generalized Center Problems with Outliers. *the ACM Transactions on Algorithms (TALG)*, 2019. Prelim. version in ICALP 2018.

[CNS22] Deeparnab Chakrabarty, Maryam Negahbani, and Ankita Sarkar. Approximation algorithms for continuous clustering and facility location problems. In *European Symposium on Algorithms*, 2022.

[CS03] Fabián A Chudak and David B Shmoys. Improved approximation algorithms for the uncapacitated facility location problem. *Journal of the ACM (JACM)*, 33(1):1–25, 2003.

[CS19] Deeparnab Chakrabarty and Chaitanya Swamy. Approximation algorithms for minimum norm and ordered optimization problems. In *the Symposium on Theory of Computing (STOC)*, pages 126–137, 2019.

[CT21] Julia Chuzhoy and Zihan Tan. Towards tight (er) bounds for the excluded grid theorem. *Journal of Combinatorial Theory, Series B*, 146:219–265, 2021.

[DLR22] Shichuan Deng, Jian Li, and Yuval Rabani. Approximation algorithms for clustering with dynamic points. *Journal of Computer and System Sciences*, 130:43–70, 2022.

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[FL11] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 569–578, 2011.

[FLNFS21] Hendrik Fichtenberger, Silvio Lattanzi, Ashkan Norouzi-Fard, and Ola Svensson. Consistent k-clustering for general metrics. In *the Symposium on Discrete Algorithms (SODA)*, pages 2660–2678, 2021.

[FS25] Sebastian Forster and Antonis Skarlatos. Dynamic consistent k-center clustering with optimal recourse. In *the Symposium on Discrete Algorithms (SODA)*, pages 212–254, 2025.

[GI25] Ameet Gadekar and Tanmay Inamdar. Dimension-Free Parameterized Approximation Schemes for Hybrid Clustering. In Olaf Beyersdorff, Michał Pilipczuk, Elaine Pimentel, and Nguyên Kim Thang, editors, *42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025)*, volume 327 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:20, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[HK56] AJ Hoffman and JB Kruskal. Integral boundary points of convex polyhedra, in "linear inequalities and related systems"(hw kuhn and aw tucker, eds.). *Annals of Mathematical Studies*, 38, 1956.

[HS86] Dorit S Hochbaum and David B Shmoys. A unified approach to approximation algorithms for bottleneck problems. *Journal of the ACM (JACM)*, 33(3):533–550, 1986.

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

[KBI19] Shrinu Kushagra, Shai Ben-David, and Ihab Ilyas. Semi-supervised clustering for de-duplication. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1659–1667. PMLR, 2019.

[KLS18] Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In *the Symposium on Theory of Computing (STOC)*, pages 646–659, 2018.

[KR07] Stavros G Kolliopoulos and Satish Rao. A nearly linear-time approximation scheme for the euclidean k-median problem. *SIAM Journal on Computing*, 37(3):757–782, 2007.

[Kus20] Shrinu Kushagra. Three-dimensional matching is np-hard. *arXiv preprint arXiv:2003.00336*, 2020.

[ŁHG+24] Jakub Łącki, Bernhard Haeupler, Christoph Grunau, Rajesh Jayaram, and Václav Rozhoň. Fully dynamic consistent k-center clustering. In *the Symposium on Discrete Algorithms (SODA)*, pages 3463–3484, 2024.

[Li13] Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 222:45–58, 2013.

[LS16] Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. *SIAM Journal on Computing (SICOMP)*, 45(2):530–547, 2016.

[LV17] Silvio Lattanzi and Sergei Vassilvitskii. Consistent k-clustering. In *the International Conference on Machine Learning (ICML)*, pages 1975–1984, 2017.

[MMR19] Konstantin Makarychev, Yury Makarychev, and Ilya Razenshteyn. Performance of johnson-lindenstrauss transform for k-means and k-medians clustering. In *the Symposium on Theory of Computing (STOC)*, pages 1027–1038, 2019.

[MV20] Sepideh Mahabadi and Ali Vakilian. (Individual) Fairness for k-Clustering. In *the International Conference on Machine Learning (ICML)*, pages 7925–7935, 2020.

[MV21] Yury Makarychev and Ali Vakilian. Approximation algorithms for socially fair clustering. In *the Conference on Learning Theory (COLT)*, 2021.

[Ple87] Ján Plesník. A heuristic for the p-center problems in graphs. *Discrete Applied Mathematics*, 17(3):263–268, 1987.

[PW01] János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. *Graphs and Combinatorics*, 17:717–728, 2001.

[RS86] Neil Robertson and Paul D Seymour. Graph minors. v. excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[Sch21] Marcus Schaefer. A new algorithm for embedding plane graphs at fixed vertex locations. *The Electronic Journal of Combinatorics*, pages P4–55, 2021.

[STA97] David B Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems. In *the Symposium on Theory of Computing (STOC)*, pages 265–274, 1997.

[Ste13] Ernst Steinitz. Bedingt konvergente reihen und konvexe systeme. *Journal für die reine und angewandte Mathematik*, 1913.

[Swa16] Chaitanya Swamy. Improved approximation algorithms for matroid and knapsack median problems and applications. *the ACM Transactions on Algorithms (TALG)*, 12(4):1–22, 2016.

[YC15] Li Yan and Marek Chrobak. Lp-rounding algorithms for the fault-tolerant facility placement problem. *Journal of Discrete Algorithms*, 33:93–114, 2015.

[Zin03] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *the International Conference on Machine Learning (ICML)*, pages 928–936, 2003.

A Useful facts

Fact 39. *For any $f(n) = n^{o(1)}$ and any variable $\tau \geq 1$, the function $(f(n))^\tau$ is bounded above by $O_\tau(1) + n$. For example, when $f(n) = \log n$, we have $(\log n)^\tau \leq 2^{\tau^2} + n$.*

Proof. Let $g(n) = \log f(n)$, and note $g(n) = o(\log n)$. If τ satisfies $k\tau \leq \log n$, then clearly $f(n)^\tau \leq n$. On the other hand, if $\tau > \frac{\log n}{k}$, then the fact that $g(n) = o(\log n)$ implies that τ grows as a function of n, and so $f(n)^\tau = O_\tau(1)$.

We now give an example that provides explicit bounds for the case $f(n) = \log n$. If $\tau \leq \log\log n$, then $(\log n)^\tau \leq 2^{(\log\log n)^2} \leq n$. Otherwise, if $\tau > \log\log n$, then $(\log n)^\tau = 2^{\tau \log\log n} \leq 2^{\tau^2}$. □

Observation 40. *Fix an aggregator function Ψ. Suppose that one can find an α-approximation to the Ψ-aggregate k-supplier (or k-median, or (k,z)-clustering) problem on T metrics and n vertices in time $f(n, T, k, \alpha)$. Then one can find an α-approximation to generalized aggregate clustering, with the restriction that weights come from $\{0,1\}$, in time $f(2n+1, T, k+1, \alpha) + \mathrm{poly}(n, T)$.*

Proof. Suppose we are given an instance η of $\{0,1\}$-weighted generalized aggregate k-clustering, defined by metrics $d_1, \ldots, d_T$, client set C, and facility set F. We construct an instance η' of vanilla aggregate $(k+1)$-clustering as follows. Create a new vertex r, and let the new facility set F' be $F' \coloneqq F \cup \{r\}$. The client set C' is set to be C. We assume WLOG that C' and F' are disjoint: if some vertex v is both a client and facility, we duplicate v, and make one copy a client and one a facility. After this duplication, we have $|F' \cup C'| \leq 2|F \cup C| + 1 = 2n + 1$.

For every $t \in [T]$, we construct a new distance metric d'_t as follows. We define the *active vertices* to be the set of clients with weight 1 in the t-th instance, together with the set of facilities excluding r (that is, $F' \setminus \{r\} = F$). The *inactive vertices* are those clients with weight 0 in the t-th instance, and the facility r. For any two vertices $v_1, v_2 \in C \cup F$, we define

$$\hat{d}'_t(v_1, v_2) \coloneqq \begin{cases} d_t(v_1, v_2) & \text{if } v_1 \text{ and } v_2 \text{ are both active} \\ \infty & \text{if one of } \{v_1, v_2\} \text{ is active and the other is inactive} \\ 0 & \text{if } v_1 \text{ and } v_2 \text{ are both inactive} \end{cases}$$

One can compute a solution to the instance η from a solution to η'. Indeed, suppose $S' \subseteq F'$ is a set of $k+1$ facilities with aggregate cost κ in η'. It is straightforward to see that, if κ' is finite, then facility r is included in S', and the set $S \coloneqq S' \setminus \{r\}$ has aggregate cost κ for the instance η. Moreover, if $S \subseteq F$ is any set of k facilities with aggregate cost κ in η, then the set of facilities $S' \coloneqq S \cup \{r\}$ has aggregate cost κ in η'. This proves the claim: in polynomial time one can construct η', and in time $f(2n+1, T, k+1, \alpha)$ one can compute an approximate solution S' and return $S' \setminus \{r\}$. □